

# Guide de conception du dossier projet

CHAOUCHÉ  
Malek

2023

ANNEE

# Rapport de Projet : XXXX

Titre Professionnel de Concepteur Développeur d'Application

NOM PRENOM

# Table des matières

I - Personal presentation .....	3
II - Présentation du projet Quizzy .....	3
III - Cahier des charges .....	4
A - Les objectifs .....	4
B - Cibles .....	4
C - Exigences fonctionnelles .....	4
1 - Fonctionnalités "Front Office" .....	4
2 - Fonctionnalités "Back Office" : Administration .....	5
3 - L'utilisateur (public) .....	6
4 - Confidentialité .....	6
5 - Droits d'accès .....	7
6 - Authentification .....	7
D - Exigences et choix techniques .....	7
1 - Exigence .....	7
2 - Choix .....	8
E - Définition du MVP (Minimum Viable Product) .....	11
F - Contrainte de temps .....	11
IV - Méthodologie et organisation .....	12
V- Conception de l'interface graphique .....	14
A - Zoning .....	14
C - Charte graphique .....	16
1 - Les couleurs .....	16
2 - Le logo .....	16
B - Wireframe .....	17
C - Maquettage .....	18
VI - Conception de la base de données .....	19
A - Modèle Conceptuel de Données (MCD) .....	19
B - Modèle Logique de Données (MLD) .....	21
VII - Conception de l'application .....	23
A - Diagramme de cas d'utilisation (Use Case) .....	23
B - Diagramme de Classe .....	25
IX - Conception Multicouche : MVC .....	27
A - Le Modèle .....	27

B – La Vue .....	28
C – Le Contrôleur .....	29
D – Communication entre nos 3 composants .....	30
E – L'architecture 3 tiers .....	31
X – Sécurité .....	32
A – Les attaques XSS (Cross-Site Scripting) .....	32
B - Les injections SQL .....	33
C – Les attaques CSRF (Cross-Site Request Forgery) .....	34
XI - Politique de test .....	36
A – Les tests unitaires .....	36
B – Les tests d'intégration .....	38
XII - Veille Technologique .....	39
A – Veille globale .....	39
B – Veille sécurité .....	40
XIII - Difficultés Rencontrées .....	42
XIV - Conclusion .....	42
XV – Remerciements .....	43
Annexe .....	44
Cahier des charges .....	45

# I - Personal presentation

Hello everyone, let me introduce myself.

My name is Bruno, I am 38 years old, I live in Sartrouville, I'm in a civil partnership and I'm the father of 2 children, an 8-years old boy and a 5-years old girl.

To say a few words about myself, I'm interested in new technologies, both hardware and software and that's one reason why I went into web development.

Now, about my background, after my BTS in industrial maintenance, I worked about nine years as a technician in air conditioning, plumbing, electricity... but I didn't feel fulfilled, that's why I started a career change in IT, especially in IT maintenance, and then after having worked in this field for about 3 years where I discovered a bit of programming, I decided to start a new career change, or rather, to specialize in development.

I have already passed a certificate as a web developer, and I wish to re-enforce my skills with this current work-study program.

Finally, the project I will present today is a community quiz application named Quizzy.

With Quizzy, users will be able to participate in quizzes but not only! I talked about "community" because users will interact with each other, by adding each other as friends and sharing their scores.

Users will also be able to create their own quizzes, questions and answers, and decide to share it (or not) with all Quizzy users or only with their friends.

**L'intro en anglais est obligatoire et doit être réalisé à l'oral le jour J**

# II - Présentation du projet Quizzy

Comme indiqué dans ma présentation personnelle juste avant, Quizzy est destinée à devenir une application Web de quiz communautaire, communautaire dans le sens où les utilisateurs de l'application seront amenés à interagir entre eux.

Pour aller un peu plus dans le détail, un utilisateur pourra créer entièrement ses propres quiz, de A à Z, titre, description et choix de la catégorie, puis ses propres questions et réponses. Il pourra aussi ajouter des amis avec lesquels il lui sera possible de participer à des quiz en simultané ou alors leur proposer de participer à des quiz de son invention. De manière un peu plus globale, Quizzy présentera donc des quiz répertoriés au sein de différentes catégories, et ses derniers seront accessibles à l'utilisation aux utilisateurs enregistrés.

De par son aspect communautaire, Quizzy a pour vocation de proposer à ses utilisateurs, le plus large choix de quiz dans des catégories aussi diverses que variées et cela, en s'appuyant sur l'imagination de ces mêmes utilisateurs.

# III - Cahier des charges

## A - Les objectifs

Comme indiqué en page 3 du cahier des charges en annexe de ce rapport, Quizzy a de multiples objectifs.

En effet, au-delà de proposer des quiz, l'objectif principal est de divertir tous les visiteurs du site et ce, quel que soit leur âge.

Comme dit dans le chapitre précédent, il est aussi nécessaire que le choix de quiz

proposé

soit le plus large possible mais aussi entièrement catégorisé, afin d'offrir au visiteur la possibilité de jouer avec un domaine qu'il maîtrise ou au contraire, de s'aventurer dans de nouvelles étendues de savoir.

Pour varier les plaisirs, les joueurs devront pouvoir profiter de différents types de questions.

De la même manière, il sera tout aussi nécessaire que chaque joueur puisse gérer (ajouter ou supprimer) les membres de sa liste d'amis à loisir.

Enfin, les utilisateurs devront pouvoir avoir accès à un tableau de leur score personnel mais aussi à un tableau des meilleurs scores par quiz (publiques).

## B - Cibles

Quizzy vise dans un premier temps la population active sur les réseaux sociaux, comprendre par-là, la population active âgée entre 18 et 40 ans environs. L'Optique étant de faire parler de Quizzy sur la place publique que représentent ses réseaux.

Bien sûr, il n'est pas souhaité que la cible se limite par l'âge, Quizzy souhaite atteindre la totalité de la population, francophone pour l'heure, anglophone à moyen terme, puis à plus long terme, d'autres langues.

## C - Exigences fonctionnelles

Ici, vous seront présentées les exigences fonctionnelles du projet, en détaillant les principales fonctionnalités du site et les droits associés à chaque profil utilisateur.

### 1 - Fonctionnalités "Front Office"

#### 1.1 Page d'accueil

La page d'accueil doit afficher tous les éléments de navigation nécessaires, ainsi que des liens directs vers des quiz choisis aléatoirement à chaque rafraîchissement de la page.

## 1.2 Menu de navigation

Un menu de navigation doit être disponible pour permettre aux utilisateurs de naviguer facilement entre les différentes rubriques du site.

## 1.3 Page "Mon compte"

La page "Mon compte" doit permettre à l'utilisateur d'accéder à toutes ses informations enregistrées sur le site et de les modifier si nécessaire. L'utilisateur doit également avoir la possibilité de supprimer son compte s'il le souhaite. Cette page doit également afficher l'historique des quiz réalisés par l'utilisateur ainsi que les scores obtenus.

## 1.4 Page "Quiz"

La page "Quiz" doit afficher trois thèmes choisis aléatoirement, avec cinq quiz sous chaque thème, également sélectionnés aléatoirement. Un lien vers une page répertoriant tous les thèmes présents sur le site doit être inclus.

## 1.5 Page "Mes Quiz et Questions"

Sur cette page, l'utilisateur doit pouvoir accéder à des outils de création de quiz et de questions. Il doit également pouvoir consulter, corriger et mettre à jour la liste de ses quiz et questions créés.

## 1.6 Page "Contact"

La page "Contact" doit contenir un formulaire permettant à l'utilisateur de contacter l'administration du site pour poser des questions, signaler d'éventuels dysfonctionnements ou proposer des améliorations.

## 1.7 Page "Plan du site"

Une page "Plan du site" doit être disponible pour offrir une vue d'ensemble de la structure du site.

## 1.8 Page "Mentions légales"

Une page "Mentions légales" doit être présente pour fournir les informations juridiques et réglementaires relatives au site.

# 2 - Fonctionnalités "Back Office" : Administration

## 2.1 Profils et droits

L'administration doit permettre la gestion de trois profils distincts, chacun ayant des droits spécifiques :

### 2.1.1 Modérateur

Les modérateurs ont pour rôle de contrôler chaque nouveau quiz ou question proposé pour éviter tout contenu choquant ou inapproprié pour les publics mineurs ou sensibles. Ils doivent également pouvoir mettre à jour les questions signalées comme ayant des formulations ou des réponses erronées. Cependant, les modérateurs ne peuvent pas supprimer de contenu. Leurs droits comprennent :

- Ajout/modification des quiz "publics"
- Ajout/modification des questions "publics"
- Ajout/modification des thèmes
- Accès aux mails reçus de la part des administrateurs

## 2.1.2 Administrateur

Les administrateurs sont responsables de la gestion globale du contenu du site. Ils ont un accès quasi total aux fonctionnalités d'administration, comprenant :

- Ajout/modification/suppression des profils Modérateur
  - Ajout/suppression de comptes utilisateurs (sur demande particulière en cas de dysfonctionnement)
- Ajout/modification/suppression des quiz "publics"

## 3 - L'utilisateur (public)

Une fois qu'un utilisateur a créé son compte, il doit pouvoir accéder au contenu du site et gérer ses informations personnelles. Les fonctionnalités disponibles pour l'utilisateur comprennent :

- Consultation/modification/suppression de son compte
- Ajout de quiz
- Consultation/modification/suppression de ses quiz "privés"
- Ajout de questions
- Consultation/modification/suppression de ses questions "privées"
- Ajout/consultation/suppression d'autres utilisateurs dans sa liste d'amis.

## 4 - Confidentialité

Il est essentiel de protéger l'accès en lecture ou en écriture aux informations "privées" de chaque utilisateur. Cependant, il doit être possible de supprimer un compte si l'utilisateur en fait la demande, par exemple en cas de dysfonctionnement. De plus, les quiz ou les questions créés par un utilisateur avec un statut privé ne doivent en aucun cas être visibles ou modifiables pour l'administration du site ou d'autres utilisateurs qui ne font pas partie de la liste d'amis du créateur du contenu.

## 5 - Droits d'accès

L'accès aux différentes fonctionnalités et informations du site sera contrôlé en fonction du profil de l'utilisateur. Les droits d'accès (lecture, écriture) doivent être déterminés en fonction de la catégorie d'informations et de l'identité de l'utilisateur.

## 6 - Authentification

Le site doit mettre en place un processus d'identification et d'authentification sécurisé des utilisateurs afin d'assurer la traçabilité de leurs actions, ainsi que l'intégrité et la non-falsification des informations.

Ces exigences fonctionnelles servent de base pour le développement du site et garantissent un fonctionnement efficace et sécurisé pour les différents utilisateurs et profils impliqués dans l'application.

## D - Exigences et choix techniques

### 1 - Exigence

Afin de pouvoir atteindre le plus largement possible les cibles du projet, il conviendra que l'application soit fonctionnelle sur les principaux navigateurs internet, à savoir : Chrome, Firefox et Edge à minima.



En plus d'être adaptée aux écrans d'ordinateur, l'application devra être responsive, avec un accent particulier sur les appareils de type smartphone. Cette décision découle de la cible spécifique de mon application, qui vise principalement la population active sur les réseaux sociaux.

En prenant en compte cette cible, il est crucial de garantir une expérience utilisateur optimale sur les smartphones, qui sont les principaux dispositifs utilisés pour accéder aux réseaux sociaux.

Bien que je ne néglige pas les autres types d'écrans, le focus sera mis sur le développement d'une interface conviviale et adaptée aux smartphones, afin de permettre une utilisation aisée et intuitive pour les utilisateurs ciblés.

## 2 - Choix

Afin de faciliter la mise en place du responsive design, on utilisera le Framework CSS Bootstrap, ce dernier sera complété par des media query selon le besoin.

D'un point de vue langages de programmation, on optera pour le Framework Symfony pour le backend, donc PHP, et Twig pour le back office, choix opéré suite au benchmark réalisé ci-après :

Popularité et communauté :

- ✓ Symfony : Symfony est l'un des Framework PHP les plus populaires et bénéficie d'une grande communauté active. Il est soutenu par SensioLabs et dispose d'une documentation complète ainsi que d'une vaste base de connaissances. Il est à précisé que SensioLabs est une agence web française, et de fait que sa communauté est aussi très largement francophone.
- ✓ Laravel : Laravel est également très populaire et bénéficie d'une large adoption dans la communauté PHP. Il dispose d'une documentation riche et d'une communauté active qui fournit un support et des ressources.
- ✓ PHP Orienté Objet : Le PHP orienté objet est la base fondamentale de nombreux Frameworks, y compris Symfony et Laravel. Il bénéficie d'une communauté solide de développeurs PHP.

Performance :

- ✓ Symfony : Symfony est connu pour sa performance élevée et sa capacité à gérer des applications web complexes. Il offre un ensemble de fonctionnalités optimisées et permet une gestion efficace des requêtes et des ressources.
- ✗ Laravel : Laravel est réputé pour sa facilité d'utilisation et son élégance, mais il peut être légèrement moins performant que Symfony dans certaines situations de haute charge.
- ✓ PHP Orienté Objet : La performance du PHP orienté objet dépendra de sa mise en œuvre et de l'efficacité de la conception du code.

Flexibilité et modularité :

- ✓ Symfony : Symfony est connu pour sa grande flexibilité et sa modularité. Il adopte une approche basée sur les composants, ce qui permet aux développeurs de choisir les éléments spécifiques dont ils ont besoin et de les combiner pour créer des applications personnalisées.
- ✗ Laravel : Laravel propose une approche plus conventionnelle avec une structure préconfigurée. Bien qu'il offre une certaine flexibilité, il peut être moins modulaire que Symfony.
- ✗ PHP Orienté Objet : La flexibilité et la modularité du PHP orienté objet dépendront de la manière dont il est utilisé et mis en œuvre dans le projet spécifique.

## Sécurité :

- ✓ ~~Symfony~~ : Symfony possède un système de sécurité robuste intégré, offrant des fonctionnalités telles que l'authentification, l'autorisation, la protection contre les attaques CSRF (Cross-Site Request Forgery) et d'autres vulnérabilités courantes.
- ✓ ~~Laravel~~ : Laravel met également l'accent sur la sécurité en fournissant des fonctionnalités intégrées pour l'authentification, l'autorisation et la protection contre les attaques.
- ✗ PHP Orienté Objet : La sécurité en PHP orienté objet dépendra de la manière dont elle est implémentée dans le projet spécifique.

## Facilité de prise en main :

- ✗ ~~Symfony~~ : Symfony a une courbe d'apprentissage initiale plus prononcée en raison de sa modularité et de sa complexité. Cependant, une fois maîtrisé, il offre une base solide pour développer des applications web évolutives et hautement personnalisables.
- ✓ ~~Laravel~~ : Laravel est souvent considéré comme plus accessible et convivial pour les débutants, avec une syntaxe expressive et des conventions intuitives.
- ✓ ~~PHP Orienté Objet~~ : La facilité de prise en main du PHP orienté objet dépendra de la connaissance et de l'expérience du développeur en programmation orientée objet.

Concernant le choix du Framework frontend de l'application, j'ai entrepris une étude approfondie qui a abouti aux résultats présentés ci-dessous. Cette analyse comparative met en lumière les avantages, les performances, la flexibilité et les fonctionnalités offertes par plusieurs Frameworks populaires :

## Popularité et communauté :

- ✓ ~~Angular~~ : Angular est un Framework frontend développé par Google et bénéficie d'une grande popularité et d'une vaste communauté de développeurs. Il dispose d'une documentation complète, de ressources en ligne abondantes et d'un support actif de la part de Google et de la communauté.
- ✓ ~~React~~ : React, développé par Facebook, est également très populaire et bénéficie d'une large adoption dans l'industrie. Il possède une vaste communauté et une documentation complète, ainsi que de nombreuses bibliothèques tierces pour étendre ses fonctionnalités.

- ✗ ~~Vue.js~~ : Vue.js est un Framework frontend en pleine croissance, avec une communauté en expansion rapide. Bien qu'il soit moins populaire que React et Angular, il gagne rapidement en popularité et dispose d'une documentation complète et d'une communauté solidaire.

## Performance :

- ✓ ~~Angular~~ : Angular est conçu pour offrir de bonnes performances, notamment grâce à son architecture basée sur la détection de changements et à sa capacité à effectuer des optimisations automatiques. Il utilise une approche de rendu côté serveur (Server Side Rendering) pour améliorer les performances initiales.

✓ React : React est réputé pour sa performance élevée en raison de sa virtual DOM et de son algorithme de réconciliation efficace. Il offre des performances optimales pour les applications avec des mises à jour fréquentes de l'interface utilisateur.

✓ Vue.js : Vue.js offre également de bonnes performances, grâce à son système de rendu réactif et à son approche de virtual DOM. Il est connu pour sa légèreté et sa réactivité.

Facilité de développement :

✗ Angular : Angular a une courbe d'apprentissage initiale plus prononcée en raison de sa complexité et de sa structure rigide. Cependant, il fournit un ensemble complet d'outils et de fonctionnalités intégrées, ce qui facilite le développement d'applications complexes et évolutives.

✓ React : React est souvent considéré comme plus facile à apprendre et à utiliser, grâce à sa syntaxe JavaScript simple et sa flexibilité. Il permet une approche modulaire et offre une facilité d'intégration avec d'autres bibliothèques ou Framework.

✓ Vue.js : Vue.js est réputé pour sa facilité d'apprentissage et sa simplicité. Sa syntaxe claire et concise en fait un choix attrayant pour les débutants et ceux qui cherchent à démarrer rapidement.

Écosystème :

✓ Angular : Angular bénéficie d'une communauté et d'un écosystème solides, avec de nombreuses bibliothèques, outils et ressources disponibles. Il est soutenu par Google, ce qui garantit un support et une évolution continu.

✓ React : React possède une grande communauté et un vaste écosystème, avec de nombreuses bibliothèques tierces et des outils complémentaires. Il est utilisé par de nombreuses grandes entreprises et bénéficie d'un support actif de la part de la communauté.

✗ Vue.js : Vue.js a une communauté en croissance rapide et un écosystème en expansion. Bien qu'il soit moins étendu que les autres Framework, il dispose d'une bibliothèque de composants officielle et de nombreuses ressources communautaires.

Bien que l'issue de ce benchmark semble indiquer que React aurait pu être un meilleur choix, le fait d'avoir déjà eu à travailler en Angular doit être pris en compte, d'autre part, le Typescript permet de sécuriser le processus de développement en imposant le typage des données et fait de ce Framework, le candidat idéal pour le front office de notre application.

**Pas besoin d'ultra détailler chaque techno, vous pouvez juste énumérer celle choisie. Ici je vous laisse les infos car je les trouve pertinentes et cela peut vous servir à mieux comprendre le rôle de chaque techno/framework**

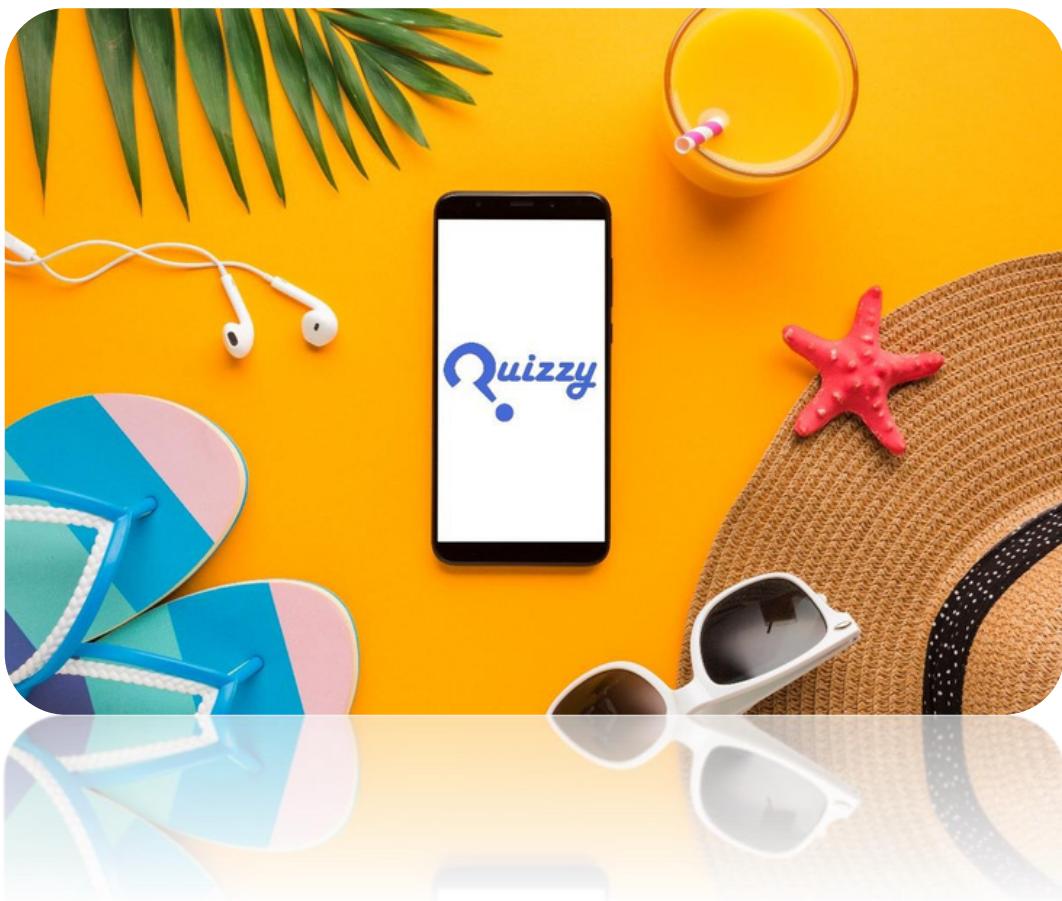
## E - Définition du MVP (Minimum Viable Product)

Dans le but d'atteindre le public cible dans un délai optimum, la possibilité de publier une première version "incomplète" de l'application est permise. Cette version devra tout de même offrir les fonctionnalités suivantes :

- Création de compte
- Participation aux quiz en solo
- Création de quiz à questions de type vrai/faux ou QCM à propositions textuelles
- Modification du profil et du mot de passe
- Une dizaine de quiz et une dizaine de catégories devront être disponibles.

## F - Contrainte de temps

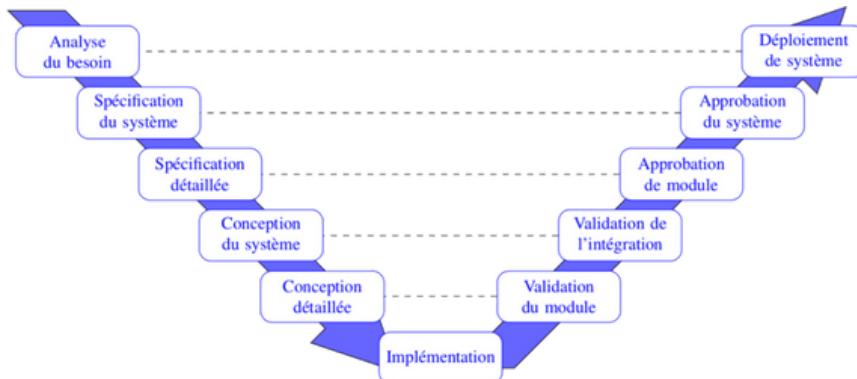
Afin de profiter de la période où nos futurs utilisateurs auront le plus de temps pour se consacrer à leur loisir, l'idéal serait de publier l'application dès les vacances scolaires début juillet, toutefois, étant donnée le démarrage tardif du projet, la publication du MVP pourra avoir lieu un peu avant le mois d'août.



## IV - Méthodologie et organisation

Dans le cadre de ce projet, J'ai fait le choix de l'utilisation de la méthode en V. La méthode en V est un modèle de développement qui met l'accent sur la séquentialité et la planification rigoureuse des activités. Elle présente plusieurs avantages dans ce contexte spécifique.

Schéma représentant les différentes étapes de la méthode en V



Tout d'abord, la méthode en V permet une meilleure compréhension et définition des besoins du projet dès le début. Vous pouvez consacrer du temps à l'analyse approfondie des exigences fonctionnelles et non fonctionnelles, ainsi qu'à la définition claire des fonctionnalités attendues.

De plus afin de m'organiser dans mes tâches et la planification de mes objectifs, j'ai utilisé l'application Miro, où j'ai mis en place un tableau s'inspirant de Kanban. En effet ce tableau permet d'avoir une vision claire sur les tâches accomplies mais aussi sur les « en-cours », celles restantes à faire et enfin les tickets à tester.



Grâce à ce tableau, j'ai notamment pu organiser mes plages de travail en démarrant ces dernières sur les tests des tickets en attentes de passage à la colonne « Done », pour passer ensuite à la sélection des tickets à traiter dans la journée.

Enfin, pour ce qui est du versioning de l'application, j'ai fait le choix de GitHub (Git). Outre son avantage pour le travail collaboratif, le fait de travailler avec GitHub offre d'une part, la disponibilité du projet d'un poste à un autre très aisément grâce au fait de disposer d'un dépôt distant, permettant ainsi, grâce à la commande git clone nom-du-dépôt, de télécharger l'intégralité du code l'application.

D'autre part, grâce au système de « versionnement » le code déjà validé (commité) est protégé et si une erreur venait à être publiée, GitHub via GIT permet le retour à une version antérieure sur laquelle on est sûr du fonctionnement de l'application.

The screenshot shows the GitHub interface for the repository 'BrunoB777 / Quizzy'. The 'Code' tab is selected, displaying the 'BackEnd' branch. The commit history for 'BackEnd' is shown in a table:

Name	Last commit message	Last commit date
..		
bin	initial commit	2 months ago
config	config api-platform	2 weeks ago
migrations	Update layout	2 weeks ago
public	Update question manage	2 weeks ago
src	Update answer manage	2 weeks ago
templates	Update answer manage	2 weeks ago
tests	initial commit	2 months ago
translations	initial commit	2 months ago
env	Update quiz manage + add api-platform and stofDoctrineExtension for g...	2 weeks ago
env.test	initial commit	2 months ago
.gitignore	initial commit	2 months ago
composer.json	Update quiz manage + add api-platform and stofDoctrineExtension for g...	2 weeks ago
composer.lock	Update quiz manage + add api-platform and stofDoctrineExtension for g...	2 weeks ago
symfony.lock	Update quiz manage + add api-platform and stofDoctrineExtension for g...	2 weeks ago

The screenshot shows the GitHub interface for the repository 'BrunoB777 / Quizzy'. The 'Branches' tab is selected, displaying the following branches:

- Default branch**: master (Updated 2 weeks ago by BrunoB777)
- Your branches**:
  - api-platform (Updated 2 weeks ago by BrunoB777)
  - dev (Updated 2 weeks ago by BrunoB777)
  - add-controller (Updated 2 weeks ago by BrunoB777)
  - add-entities (Updated 2 months ago by BrunoB777)
- Active branches**:
  - api-platform (Updated 2 weeks ago by BrunoB777)
  - dev (Updated 2 weeks ago by BrunoB777)
  - add-controller (Updated 2 weeks ago by BrunoB777)
  - add-entities (Updated 2 months ago by BrunoB777)

# V- Conception de l'interface graphique

La conception de l'interface utilisateur (UI) et de l'expérience utilisateur (UX) occupe une place primordiale dans le développement d'une application. Ce chapitre se concentre sur cette phase de conception, où nous avons accordé une attention particulière à l'aspect visuel, à l'ergonomie et à l'expérience globale des utilisateurs.

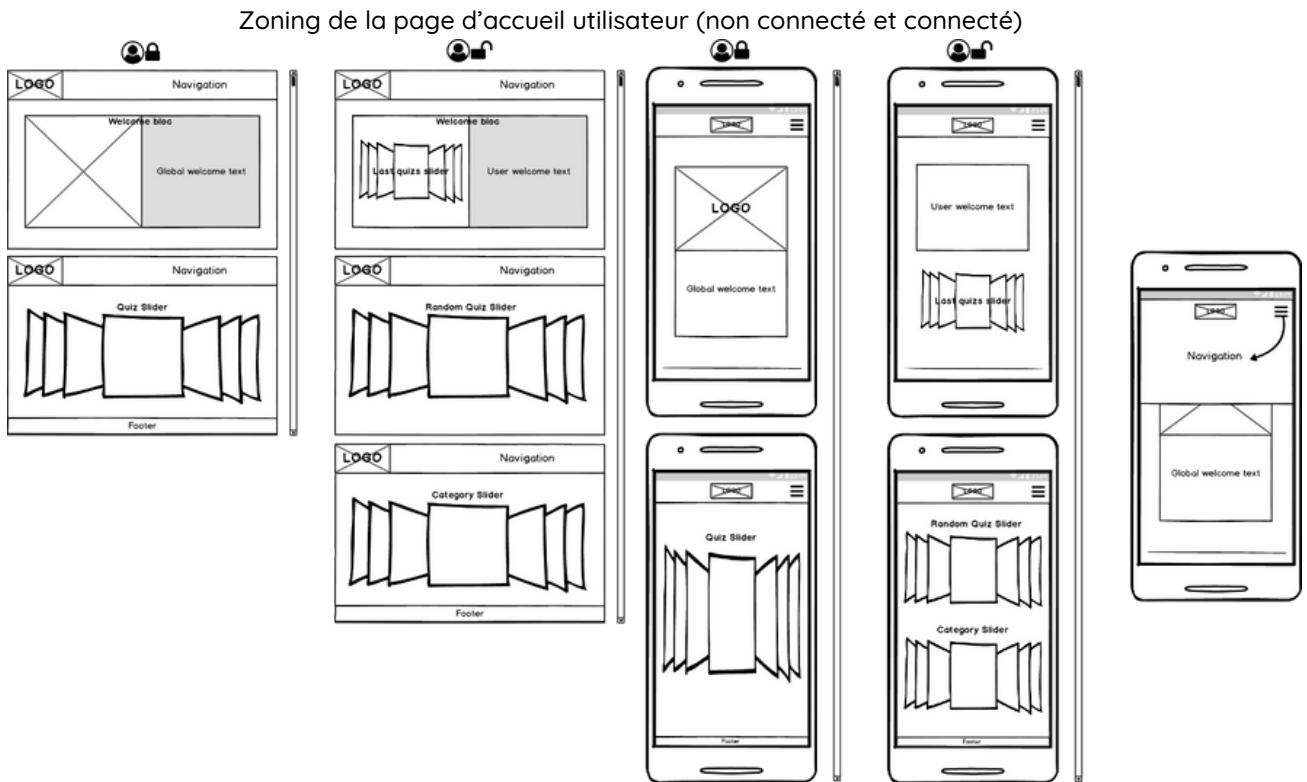
L'objectif principal de la conception UI/UX de Quizzy est de créer une interface intuitive, attrayante et conviviale, qui offre une expérience utilisateur agréable et engageante. Je me suis efforcé de concevoir des interactions fluides et des fonctionnalités intuitives qui répondent aux besoins du public cible, tout en reflétant l'identité de mon projet.

## A - Zoning

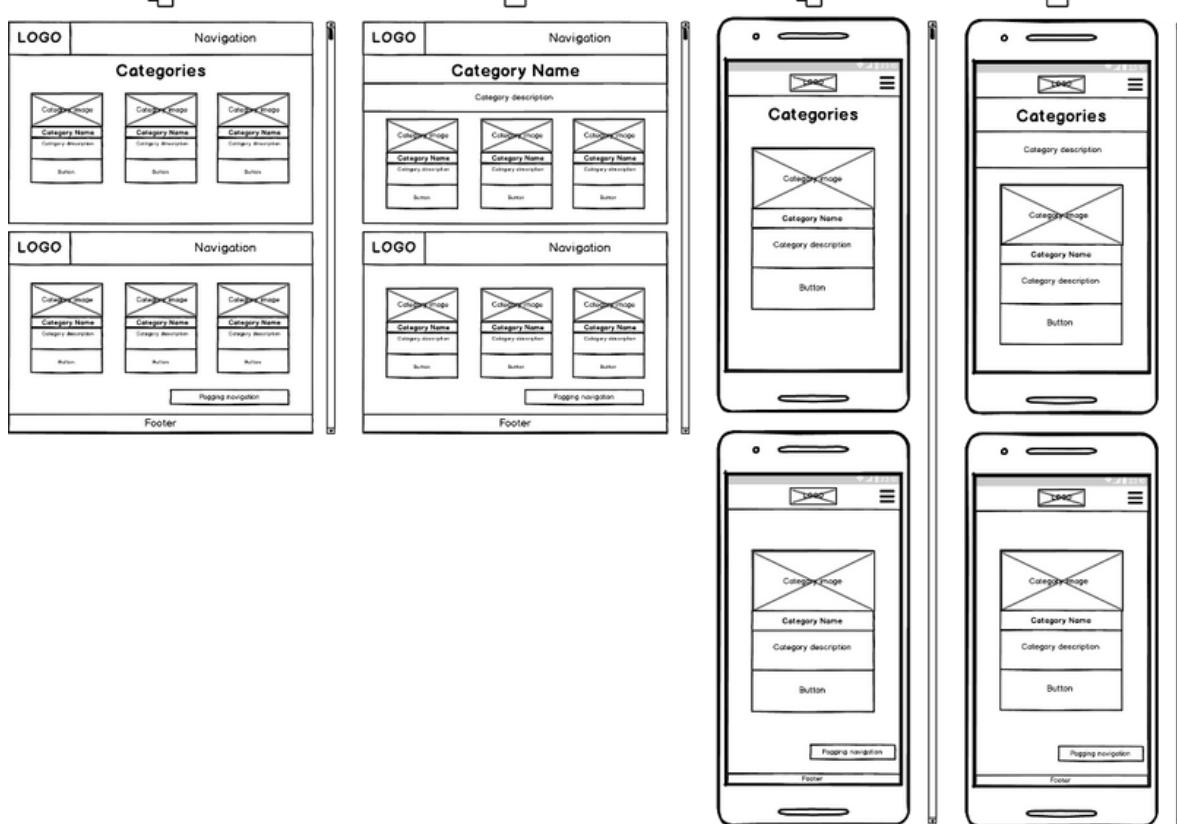
Le zoning fait référence à un processus qui consiste à représenter l'interface utilisateur de façon schématisé de manière à faire apparaître les zones qui contiendront chacunes, un élément de l'interface.

Ce procédé a pour objectif d'organiser de manière claire et structurée les éléments de l'interface utilisateur. En divisant l'écran en zones dédiées, il devient plus facile pour les utilisateurs de comprendre et de naviguer dans l'application.

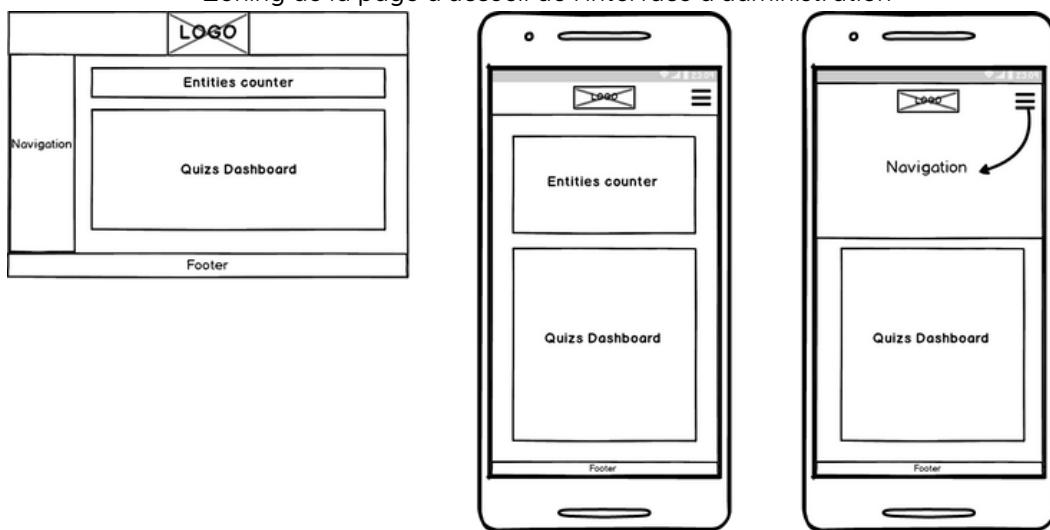
Le zoning permet aussi de mettre en évidence la hiérarchie de l'information en fonction de son importance.



## Zoning de la page affichant les catégories et d'une page n'affichant qu'une seule catégorie



## Zoning de la page d'accueil de l'interface d'administration



Pour rappel, les étapes de conception graphique sont :Zoning -> wireframe -> maquettes

## C - Charte graphique

Afin de m'aider à avancer dans le processus de création de mon interface utilisateur, j'ai pris la décision de me lancer dans l'élaboration de ma charte graphique, le but étant de me permettre d'avoir des éléments pour visualiser l'interface que je souhaiterai développer pour mon application.

### 1 - Les couleurs

L'élément qui selon moi paraissait le plus simple à déterminer a été le choix de la couleur principale. En effet, l'idée de faire de Quizzy, une application « communautaire » me donnait déjà une idée assez précise de cet élément. Je me suis donc inspiré de réseaux sociaux, Facebook, twitter, LinkedIn, etc...

- Couleur principale : #4768d5

Pour la couleur secondaire, j'ai naturellement eu pour idée de prendre une couleur claire pour qu'elle puisse ressortir sur les fonds plus sombres, mais aussi pour qu'elle puisse elle-même servir de fond à la couleur principale

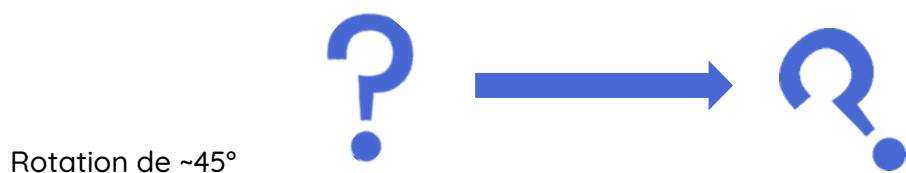
- Couleur secondaire : #cfddff

### 2 - Le logo

Pour ce qui est du logo, Quizzy étant une application de quiz, j'ai rapidement pensé à utiliser un « ? », et j'ai rapidement su comment j'allais l'intégrer.

En effet, en lui faisant faire une rotation 45° environ dans le sens anti-horaire, on se retrouve avec un « Q » dont le seul défaut est que le rond qui le compose n'est pas fermé.

Toutefois, pour que le rendu reste compréhensible, toutes les polices ne sont pas adaptées, et après plusieurs essais, j'ai choisi la police « Rockwell » (police disponible dans Photoshop).



Pour le reste du logo, j'ai choisi une police qui me semblait lisible, mais qui avait aussi, un aspect rappelant l'écriture manuscrite, avec des boucles par exemple, et la police Harlow Solid Italic de Office, s'avère être très adaptée à ces contraintes. Et associé au « Q » réalisé précédemment, on obtient :



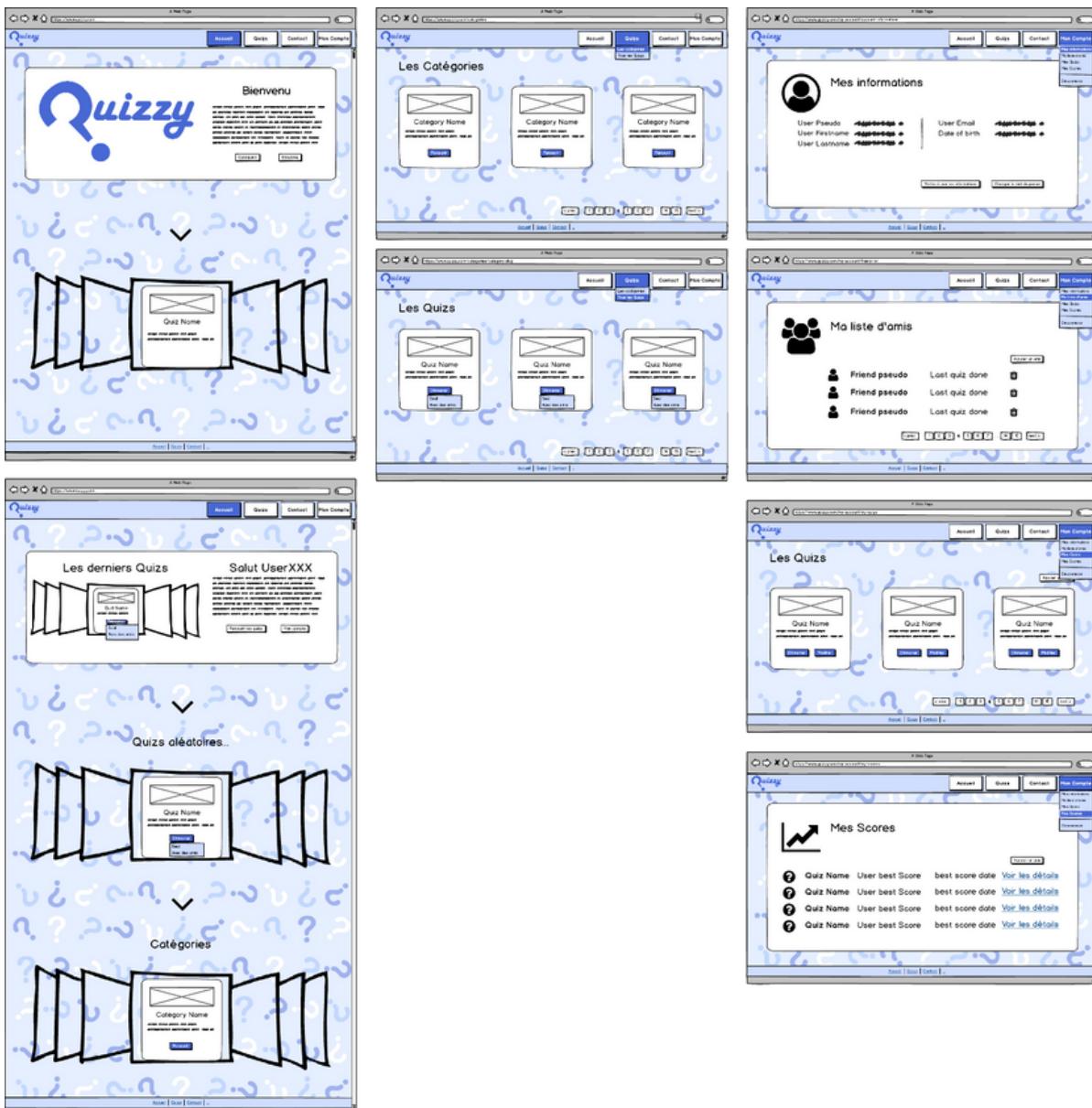
## B - Wireframe

Le wireframe est une représentation visuelle simplifiée de l'interface utilisateur, toutefois elle rentre plus en détails dans les éléments déterminés lors du zoning. Il s'agit essentiellement d'une esquisse ou d'un schéma qui met en évidence la structure, la disposition et les interactions de base des éléments de l'interface.

Ce schéma permet de définir la structure de l'application en identifiant les différentes sections, les zones d'interaction et les relations entre les éléments. Il met en évidence la disposition générale de l'interface, comme la navigation, les menus, les formulaires, etc.

En identifiant et en résolvant les problèmes d'interface dès l'édition du wireframe, on économise du temps de développement en évitant d'avoir à faire des retouches à tattons.

Wireframe de l'interface utilisateur



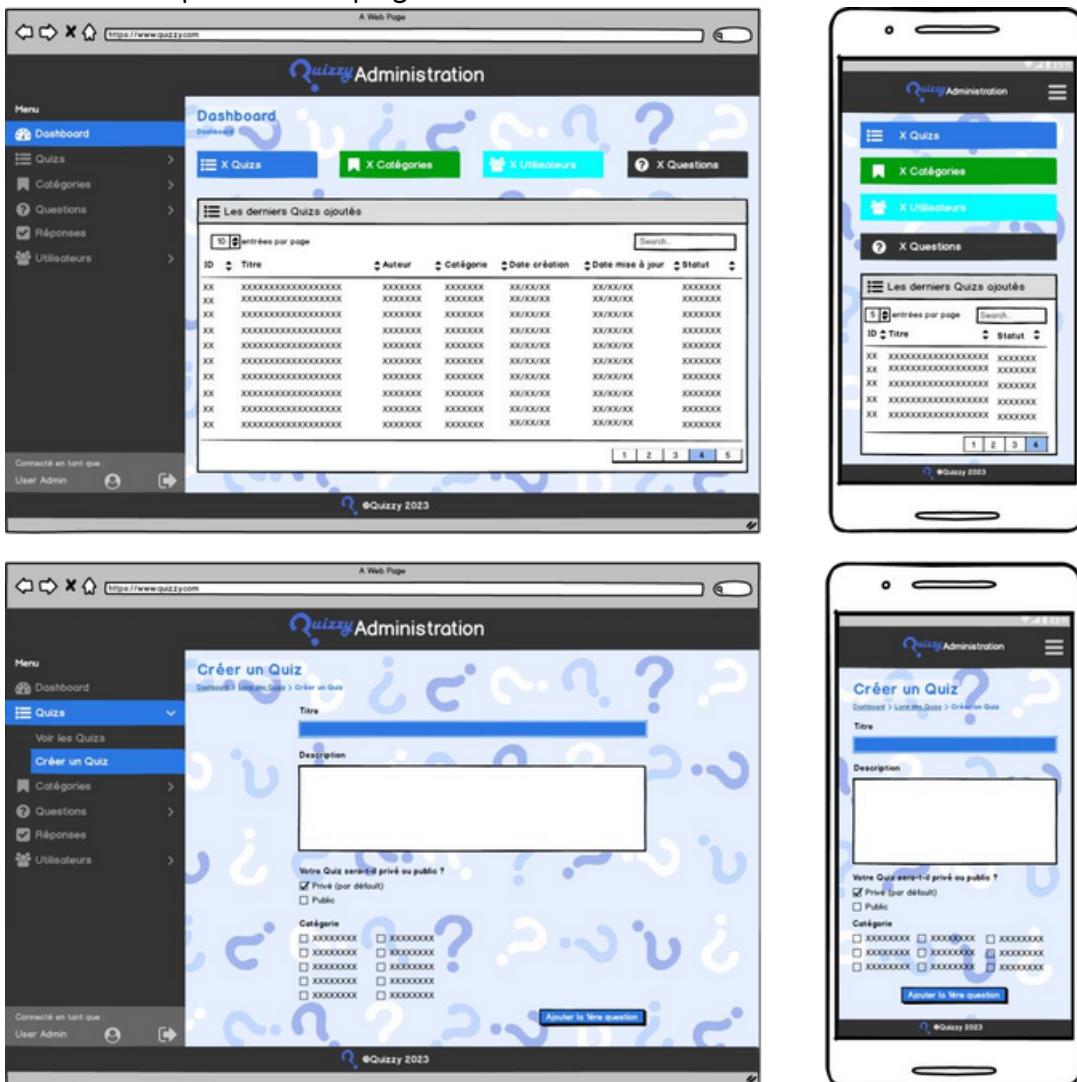
## C - Maquettage

Le maquettage fait référence à la création de maquettes graphiques détaillées de l'interface utilisateur. Il s'agit d'une représentation visuelle plus élaborée et esthétique de l'application, qui inclut des éléments de design tels que les couleurs, les typographies, les icônes, etc.

Les maquettes permettent une visualisation précise de l'apparence et du ressenti de l'interface utilisateur. Elles donnent une idée réaliste de l'aspect final de l'application, en intégrant les éléments visuels et les détails de conception.

Elles servent de référence visuelle pour les développeurs lors de la création de l'interface

utilisateur. Elles définissent les styles, les mises en page et les interactions attendues, ce qui facilite le processus de développement en réduisant les ambiguïtés et les erreurs potentielles. Maquette de la page d'accueil de l'interface d'Administration



# VI - Conception de la base de données

Une étude visant à structurer la base de données d'une application n'est pas un processus simple mais il est nécessaire si on ne souhaite pas développer à l'aveugle. En effet, réaliser cette étude aide à comprendre les données qui seront utilisées par l'application. Cela permet de modéliser et de représenter les entités, les relations et les attributs clés nécessaires pour le fonctionnement de l'application.

Pour réaliser cette étude, j'ai fait usage de la méthode Merise, il s'agit d'une méthode de conception et de modélisation des systèmes d'information. La méthode Merise se compose de plusieurs étapes clés, notamment l'analyse des besoins, la modélisation conceptuelle des données (MCD), la modélisation logique des données (MLD) et la modélisation physique des données (MPD). La méthode Merise permet une conception rigoureuse des bases de données.

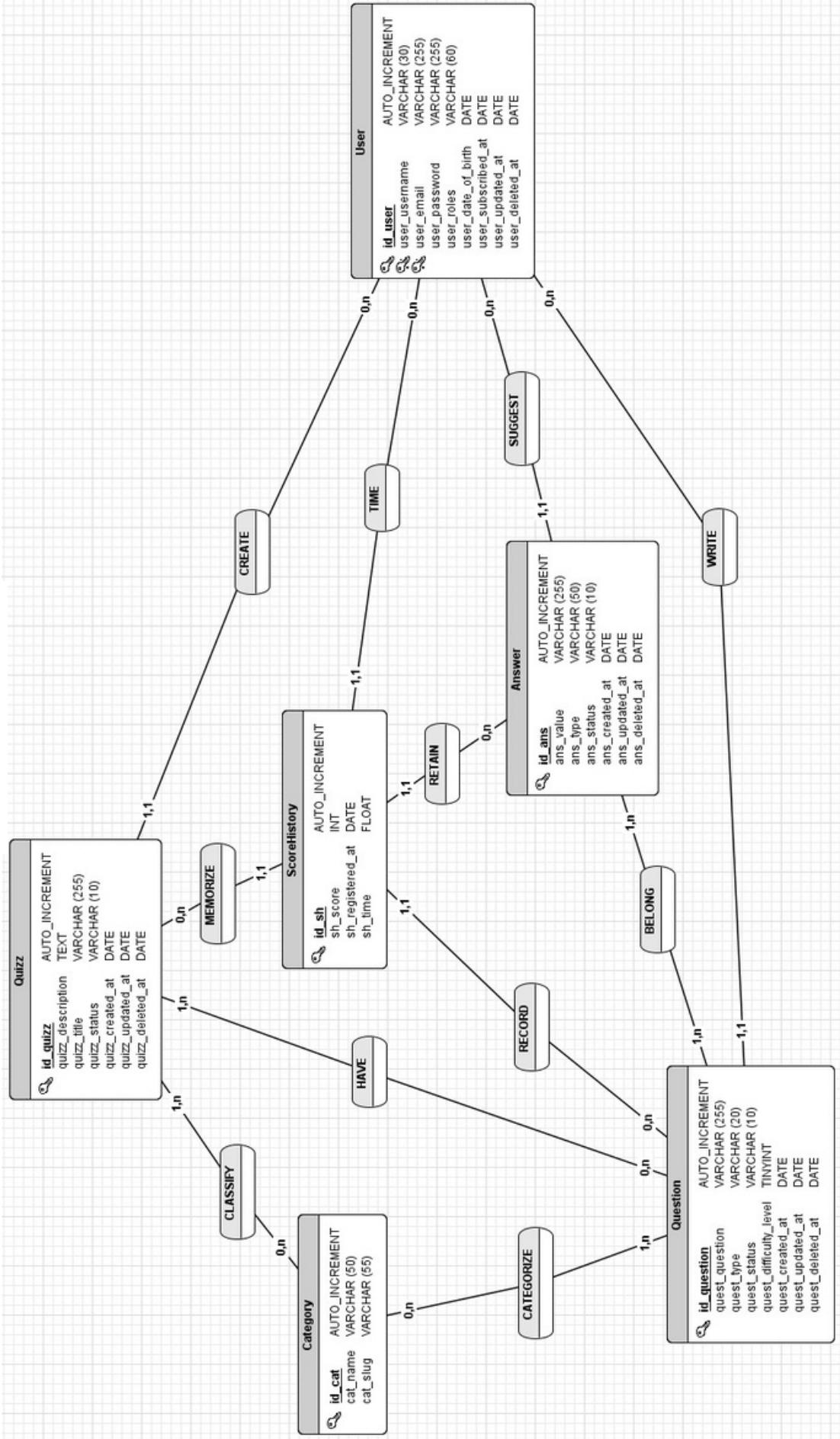
## A - Modèle Conceptuel de Données (MCD)

Le MCD, ou Modèle Conceptuel de Données, est une représentation abstraite des concepts et des relations entre les entités d'un système d'information. Il s'agit d'un schéma conceptuel qui décrit les principales entités, leurs attributs et les associations entre elles. Le MCD est indépendant de tout aspect technique ou de mise en œuvre.

Il utilise des symboles graphiques tels que des rectangles pour représenter les entités et des lignes pour représenter les relations entre ces entités. Les attributs des entités sont spécifiés à l'intérieur des rectangles, et les cardinalités des relations sont indiquées avec des notations appropriées.

Son objectif est de fournir une vue claire et abstraite des données du système, en mettant l'accent sur la structure logique et les relations entre les entités. Il permet de comprendre les besoins métier, d'identifier les entités clés, d'organiser les données de manière cohérente et de préparer la modélisation logique ultérieure.

## MCD de l'application Quizzy



## B - Modèle Logique de Données (MLD)

Le MLD, ou Modèle Logique de Données, est, quant à lui, une représentation structurée et formelle des données d'un système d'information. Il s'agit d'une étape intermédiaire entre le MCD (Modèle Conceptuel de Données) et le MPD (Modèle Physique de Données).

Il se concentre sur la traduction du MCD en un modèle plus détaillé et spécifique, en utilisant des concepts et des notations propres à un système de gestion de base de données (SGBD) particulier. Il inclut des éléments tels que les tables, les clés primaires, les clés étrangères, les contraintes d'intégrité et les relations entre les tables.

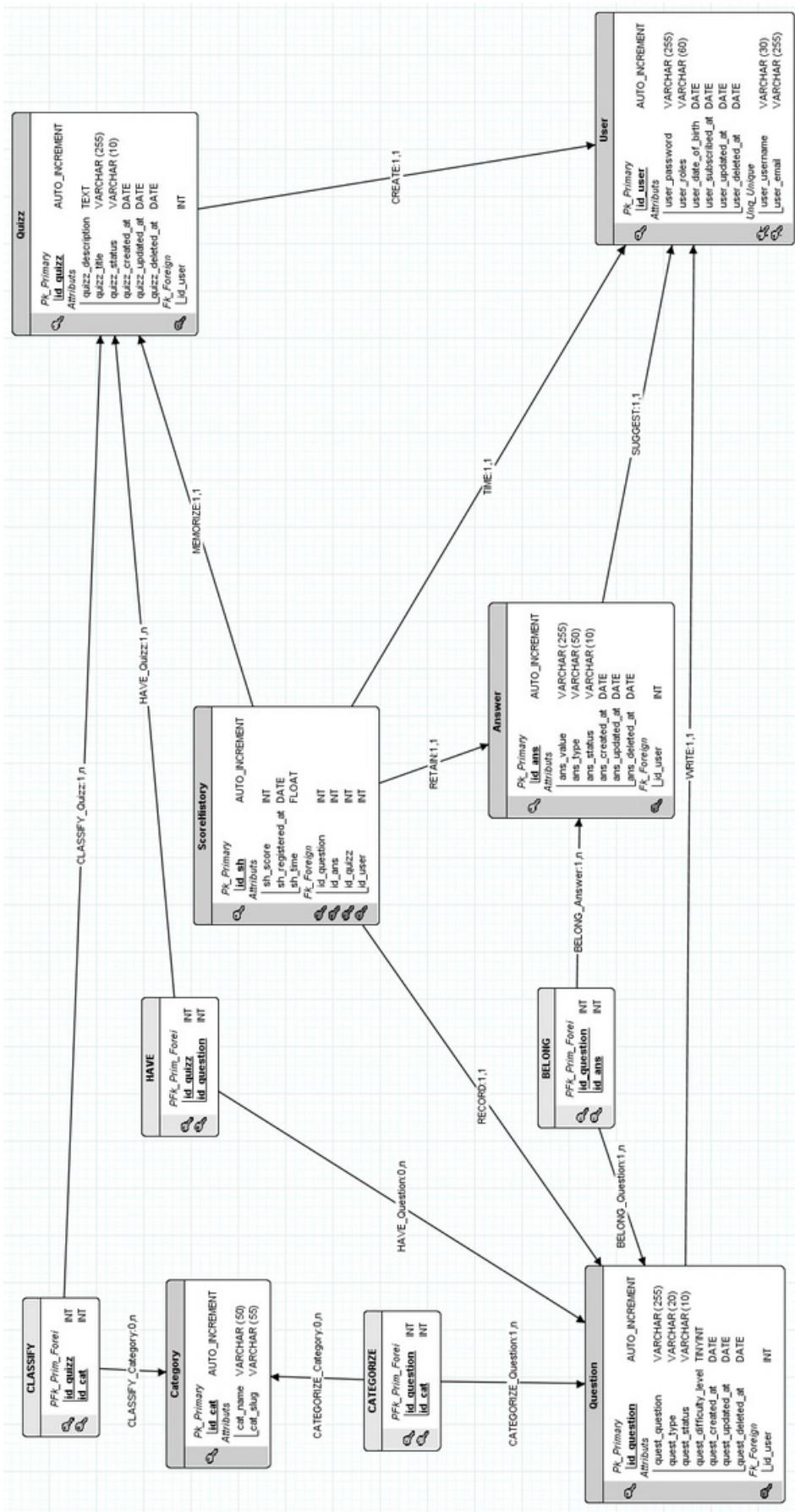
Ce schéma décrit en effet la structure logique des données, en définissant les entités comme des tables et en spécifiant les attributs de chaque table. Il identifie également les relations entre les tables en utilisant des clés étrangères pour établir des liens entre les enregistrements.

Pour l'exam, seul le MCD suffit, en revanche, il est impératif de pouvoir passer du MCD au MPD à la main.

Le jury pourra vous demander de dessiner la relation entre 2 entités du MCD vers le MPD.

Il est impératif de connaître par cœur les règles de passe du MCD au MPD.

## MLD de l'application Quizzy



## VII - Conception de l'application

Pour la conception de Quizzy, je me suis appuyé sur le langage UML (Unified Modeling Language) pour représenter le fonctionnement de l'application.

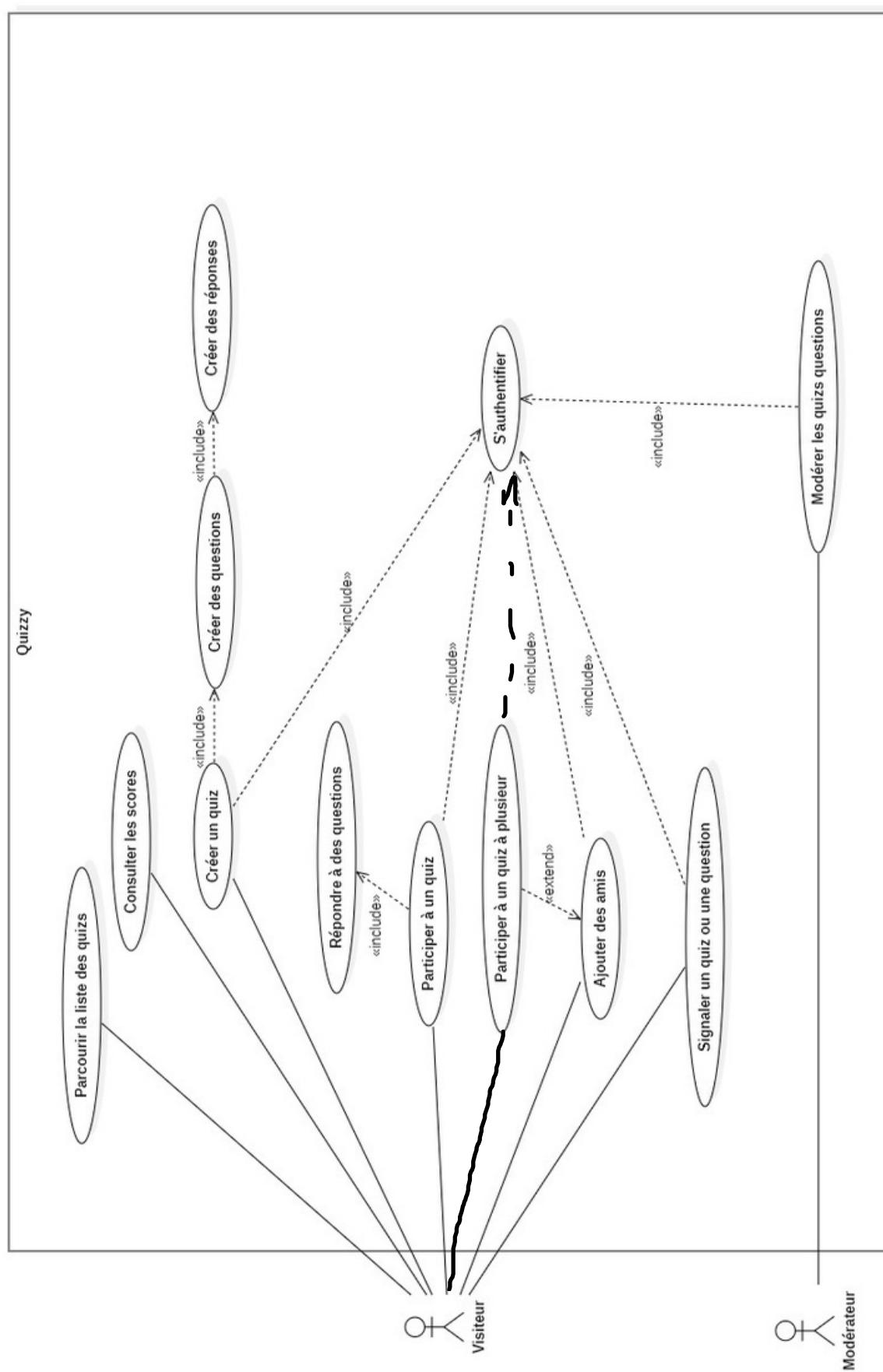
En effet, le langage UML est un langage graphique standard utilisé pour modéliser et représenter visuellement les systèmes logiciels. Il permet de capturer les aspects essentiels d'un système, de sa structure à son comportement, en utilisant une notation graphique compréhensible par les développeurs, les concepteurs et les parties prenantes. UML offre un ensemble de diagrammes qui permettent de décrire différents aspects d'un système, tels que les cas d'utilisation, les classes, les séquences, les activités, les composants, les déploiements, etc. Ces diagrammes servent de supports de communication pour analyser, concevoir, documenter et visualiser un système logiciel, facilitant ainsi la compréhension et la collaboration entre les membres de l'équipe de développement et les parties prenantes. Le langage UML est largement utilisé dans l'industrie du développement logiciel pour améliorer la modélisation, la conception et la documentation des systèmes complexes.

Parmi les schémas UML que j'ai élaborés, deux d'entre eux jouent un rôle essentiel dans la compréhension des fonctionnalités et de la structure de mon système : le diagramme de cas d'utilisation (use case) et le diagramme de classes. Ces schémas UML offrent une vision claire et structurée de mon application, permettant ainsi de visualiser les interactions entre les utilisateurs et le système, ainsi que la structure et les relations entre les différentes entités.

### A - Diagramme de cas d'utilisation (Use Case)

Le diagramme de cas d'utilisation met en évidence les différentes actions qu'un utilisateur peut effectuer et les fonctionnalités à mettre en place dans mon application. Il est essentiel pour comprendre les besoins des utilisateurs et définir les fonctionnalités de mon système en se concentrant sur leur point de vue. Il permet de capturer les exigences fonctionnelles de manière claire et organisée, en mettant l'accent sur les actions effectuées par les utilisateurs et les résultats attendu.

## Diagramme de cas d'utilisation de Quizzy



Pour le use case, il est important de bien comprendre 'include' et 'extend' et savoir l'expliquer avec vos mots

## B – Diagramme de Classe

Le diagramme de classe permet de visualiser les classes, leurs attributs, leurs méthodes et leurs relations dans un système logiciel.

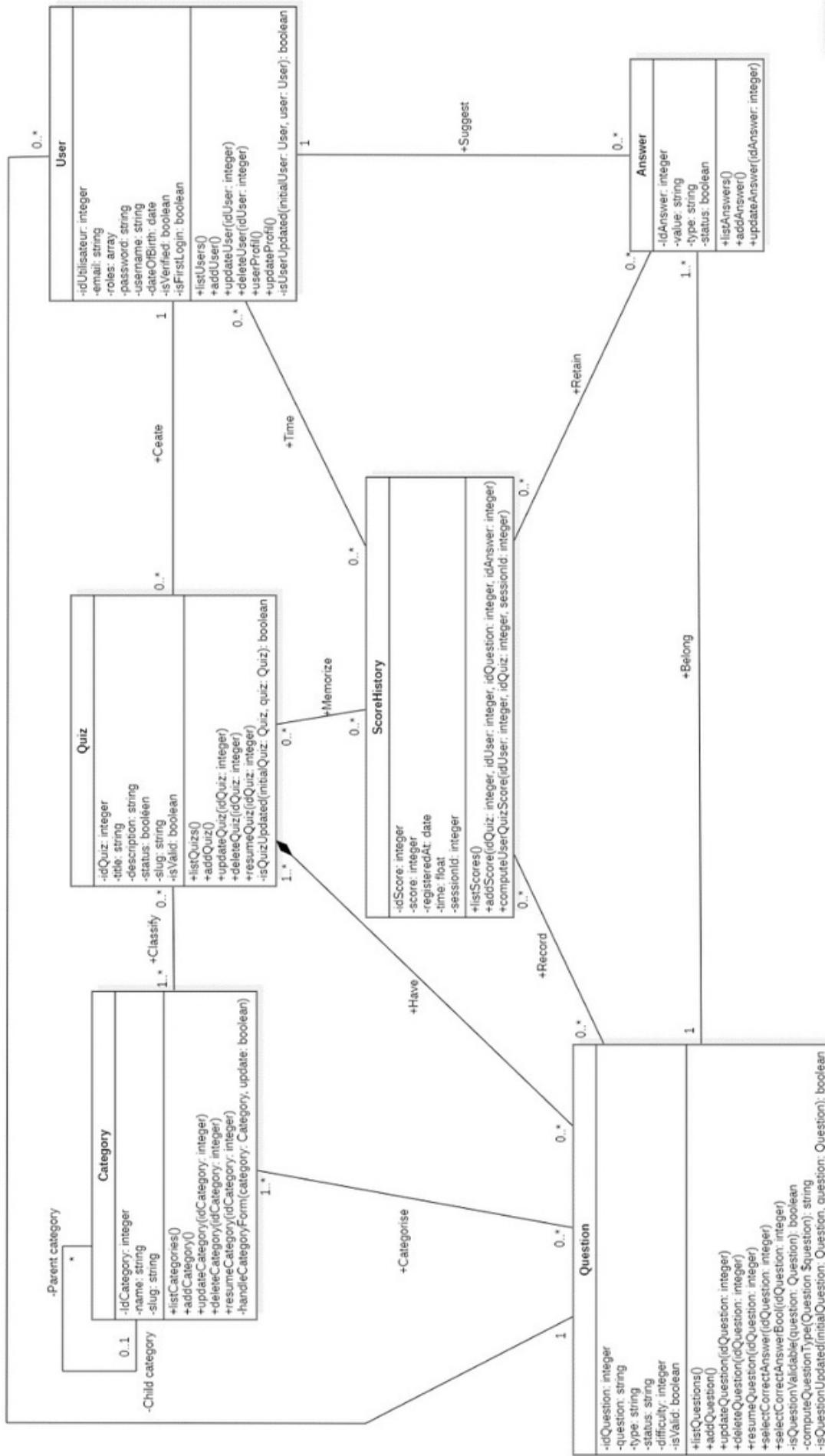
Il offre une vue d'ensemble de mon système en identifiant les différentes entités et en décrivant leurs caractéristiques et leurs relations.

Il permet de capturer les informations essentielles sur les classes, telles que leurs attributs et leurs méthodes, et de représenter la hiérarchie et les associations entre les classes.

Ce diagramme m'a permis de concevoir une structure solide pour mon application, en identifiant les entités clés, leurs attributs et leurs relations. Le diagramme de classe a facilité la compréhension de l'architecture de mon système.

**Vous pouvez choisir entre diagramme de classe ou Merise.Pas d'obligation de faire les 2.Grande recommandation pour la méthode MERISE**

## Diagramme de classe de Quizzy



# IX - Conception Multicouche : MVC

Le schéma MVC (Modèle-Vue-Contrôleur) est un concept fondamental en développement logiciel, largement utilisé dans la conception d'applications web et d'autres systèmes interactifs. Il offre une structure claire et organisée pour séparer les différentes responsabilités d'une application et favoriser une meilleure maintenabilité, extensibilité et réutilisabilité du code. Je vais ci-dessous vous présenter les composants de ce schéma dans le cadre du Framework Symfony.

## A - Le Modèle

Le Modèle représente la couche responsable de la gestion des données et de la logique métier. Il est chargé d'interagir avec la base de données, de récupérer et de stocker les informations pertinentes pour mon application.

En utilisant Symfony, j'ai pu bénéficier de l'ORM (Object Relational Mapping) Doctrine, qui facilite la création et la manipulation des entités en se basant sur les modèles de données définis. L'ORM, ici Doctrine, sert d'intermédiaire entre les classes PHP et les tables de ma base de données SQL.

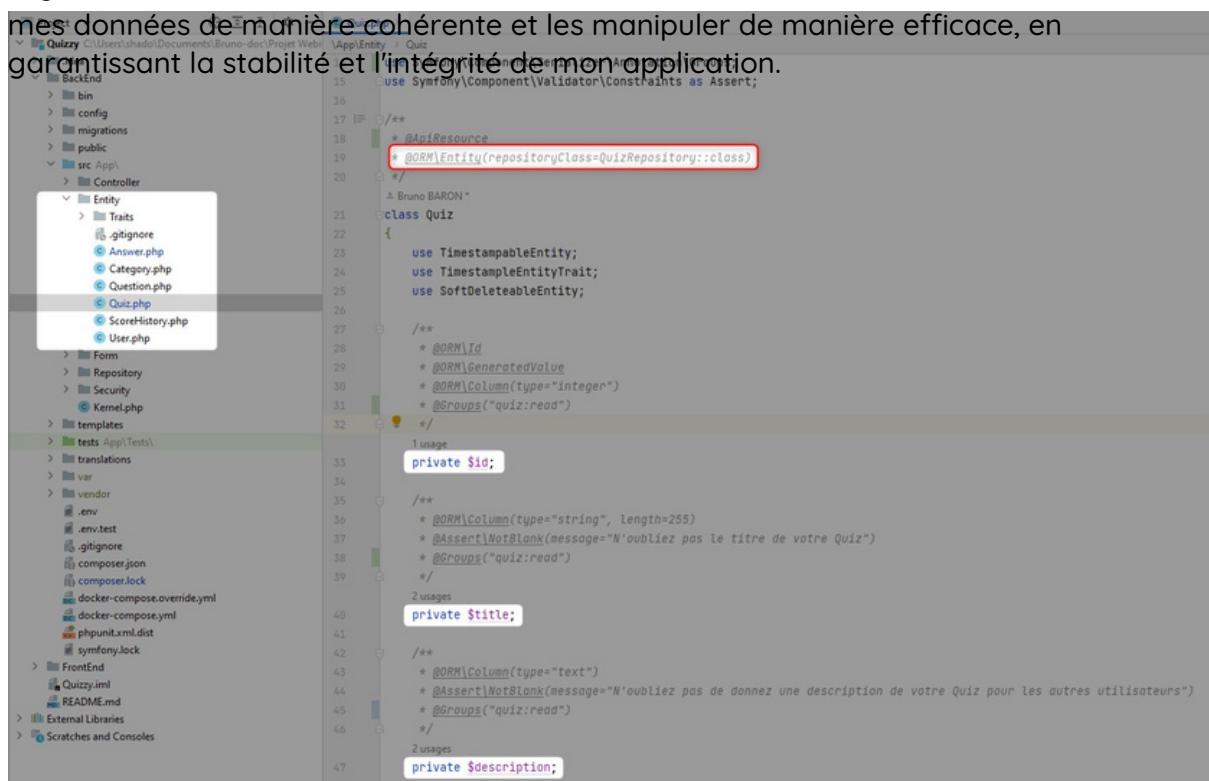
Le Modèle dans Symfony se compose donc de mes entités et de leurs relations, ainsi

que

des classes de gestion des requêtes et des opérations de validation et de persistance des données. Grâce à la clarté et à la flexibilité du Modèle dans Symfony, j'ai pu

organiser

mes données de manière cohérente et les manipuler de manière efficace, en garantissant la stabilité et l'intégrité de mon application.



The screenshot shows a file browser interface with a sidebar on the left displaying the project structure of a Symfony application named 'Quizzy'. The 'src/AppBundle/Entity' directory is selected, showing files like Answer.php, Category.php, Question.php, Quiz.php, ScoreHistory.php, and User.php. Below this, the 'Quiz.php' file is open in the main editor area. The code defines an entity 'Quiz' with annotations for ORM mapping and validation. A specific annotation, '@ORM\Entity(repositoryClass=QuizRepository::class)', is highlighted with a red box. The code also includes validation annotations like @Assert\NotBlank and @Groups.

```
use Symfony\Component\Validator\Constraints as Assert;
use Doctrine\ORM\Mapping as ORM;

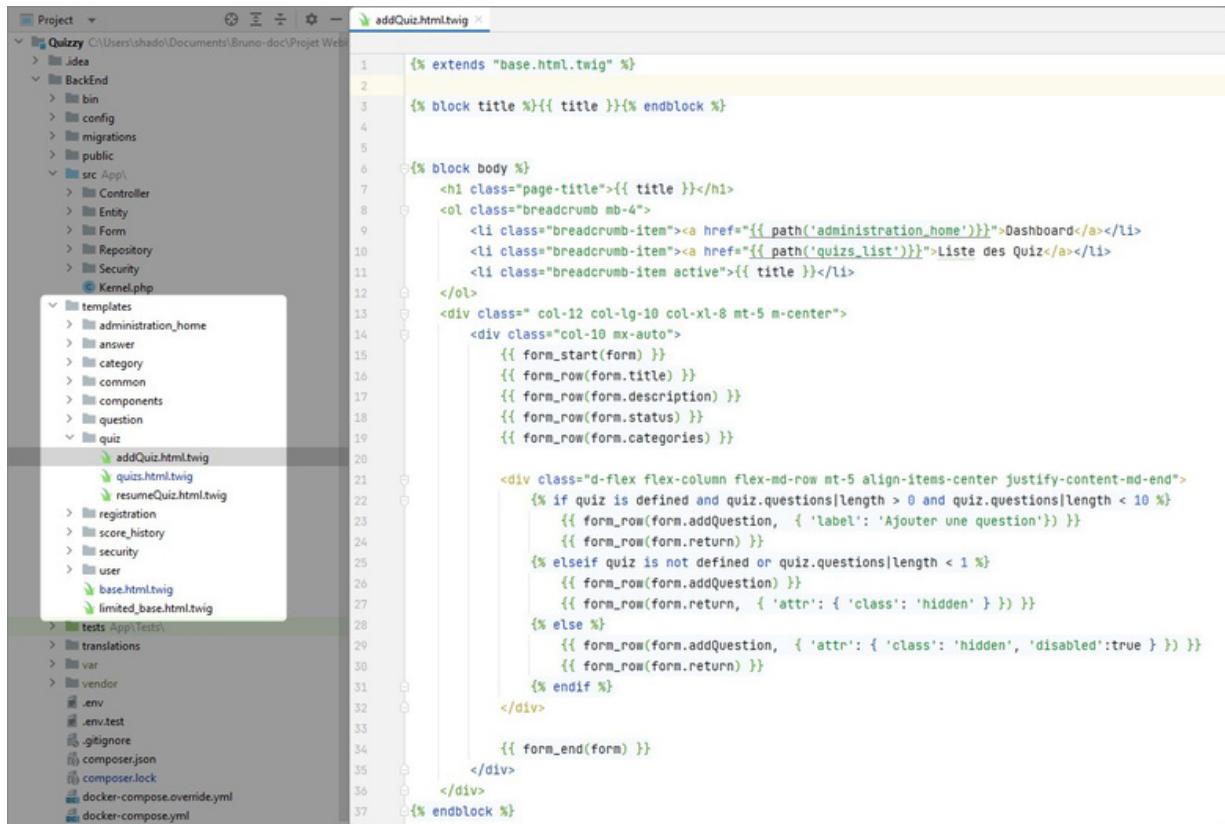
/**
 * @ORM\Entity(repositoryClass=QuizRepository::class)
 */
class Quiz
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue
     * @ORM\Column(type="integer")
     * @Groups("quiz:read")
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=255)
     * @Assert\NotBlank(message="N'oubliez pas le titre de votre Quiz")
     * @Groups("quiz:read")
     */
    private $title;

    /**
     * @ORM\Column(type="text")
     * @Assert\NotBlank(message="N'oubliez pas de donner une description de votre Quiz pour les autres utilisateurs")
     * @Groups("quiz:read")
     */
    private $description;
}
```

## B - La Vue

La Vue est responsable de la présentation des informations aux utilisateurs et de l'interaction avec eux. Grâce au moteur de templates Twig fourni par Symfony, j'ai pu concevoir des vues dynamiques en utilisant des modèles réutilisables et des composants. Les vues dans Symfony sont construites à partir des données fournies par le Contrôleur et sont responsables de l'affichage des résultats de manière structurée et conviviale. J'ai pu créer des interfaces utilisateur esthétiques et réactives en utilisant les fonctionnalités de Twig telles que les boucles, les conditions et les filtres. De plus, Symfony offre des outils pratiques pour gérer les formulaires et les interactions utilisateur, ce qui facilite la collecte de données et les actions de l'utilisateur au sein de l'application.



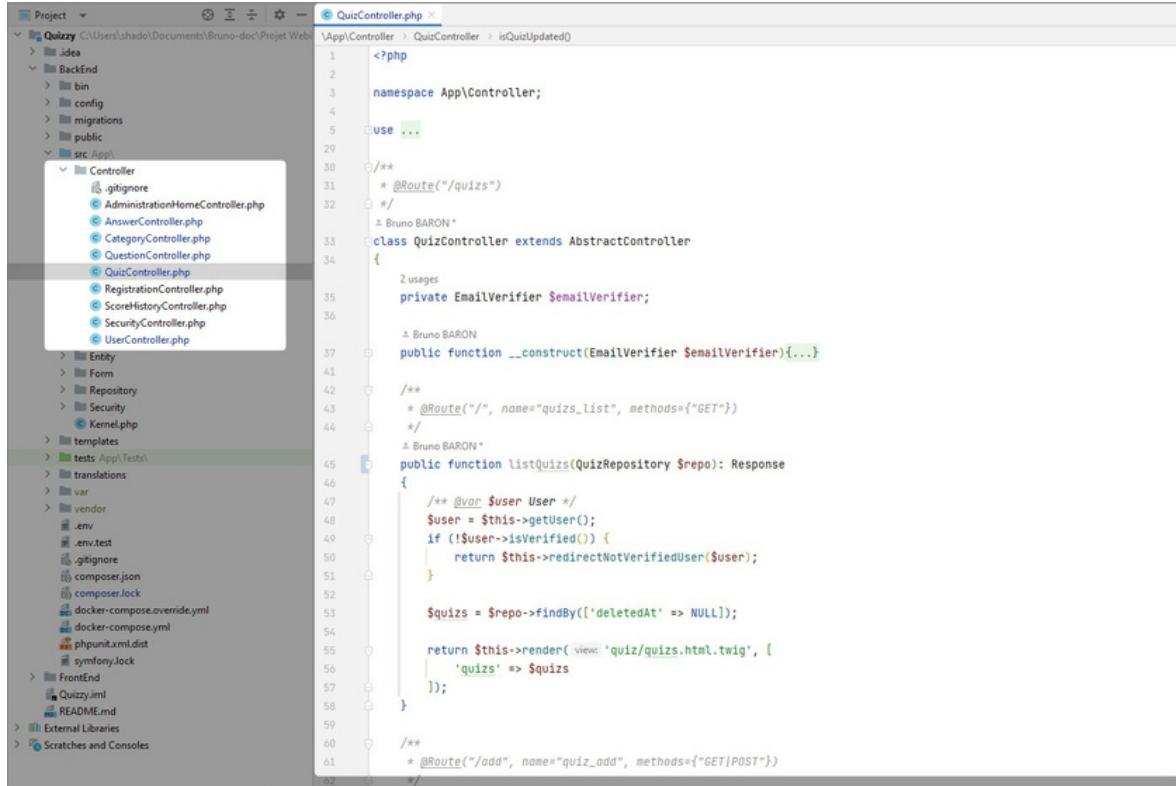
```
% extends "base.html.twig"
%
{# block title %}{{ title }}{{ endblock }}

{# block body %}
<h1 class="page-title">{{ title }}</h1>
<ol class="breadcrumb mb-4">
    <li class="breadcrumb-item"><a href="{{ path('administration_home') }}">Dashboard</a></li>
    <li class="breadcrumb-item"><a href="{{ path('quizzes_list') }}">Liste des Quiz</a></li>
    <li class="breadcrumb-item active">{{ title }}</li>
</ol>
<div class="col-12 col-lg-10 col-xl-8 mt-5 m-center">
    <div class="col-10 mx-auto">
        {{ form_start(form) }}
        {{ form_row(form.title) }}
        {{ form_row(form.description) }}
        {{ form_row(form.status) }}
        {{ form_row(form.categories) }}

        <div class="d-flex flex-column flex-md-row mt-5 align-items-center justify-content-md-end">
            {% if quiz is defined and quiz.questions|length > 0 and quiz.questions|length < 10 %}
                {{ form_row(form.addQuestion, { 'label': 'Ajouter une question' }) }}
                {{ form_row(form.return) }}
            {% elseif quiz is not defined or quiz.questions|length < 1 %}
                {{ form_row(form.addQuestion) }}
                {{ form_row(form.return, { 'attr': { 'class': 'hidden' } }) }}
            {% else %}
                {{ form_row(form.addQuestion, { 'attr': { 'class': 'hidden', 'disabled':true } }) }}
                {{ form_row(form.return) }}
            {% endif %}
        </div>
        {{ form_end(form) }}
    </div>
</div>
{{ endblock }}
```

## C - Le Contrôleur

Le Contrôleur joue un rôle crucial dans la gestion des actions de l'utilisateur et l'orchestration des interactions entre le Modèle et la Vue. Dans mon projet Symfony, le Contrôleur agit comme un intermédiaire entre les requêtes entrantes et les opérations à effectuer. Il reçoit les données saisies par l'utilisateur sous forme de requête et les transmet au Modèle pour effectuer des opérations telles que la création, la modification ou la suppression de données. Une fois que le Modèle a effectué les opérations nécessaires, le Contrôleur met à jour la Vue pour refléter les changements et renvoie la réponse appropriée à l'utilisateur. Symfony fournit un système de routage qui associe les URL aux actions des Contrôleurs, facilitant ainsi la gestion des différentes fonctionnalités de mon application. Grâce au Contrôleur, on peut donc coordonner les flux de données, gérer les erreurs et garantir la cohérence entre les différentes parties de l'application.



The screenshot shows a code editor with the file `QuizController.php` open. The file is located in the `src/App/Controller` directory of a Symfony project named `Quizzzy`. The code is a PHP class definition for `QuizController`, which extends `AbstractController`. It includes annotations for routes and security, and a constructor that injects an `EmailVerifier`. The `listQuizzes` method retrieves quizzes from a repository and renders a Twig template. The code editor interface shows the project structure on the left and the code content on the right.

```
<?php  
namespace App\Controller;  
  
use ...  
  
/** * @Route("/quizzes") */  
class QuizController extends AbstractController {  
    private EmailVerifier $emailVerifier;  
  
    public function __construct(EmailVerifier $emailVerifier) {...}  
  
    /** * @Route("/", name="quizzes_list", methods={"GET"}) */  
    public function listQuizzes(QuizRepository $repo): Response {  
        /** @var $user User */  
        $user = $this->getUser();  
        if (!$user->isVerified()) {  
            return $this->redirectToRoute('not_verified');  
        }  
  
        $quizzes = $repo->findBy(['deletedAt' => null]);  
  
        return $this->render('quiz/quizzes.html.twig', [  
            'quizzes' => $quizzes  
        ]);  
    }  
    /** * @Route("/add", name="quiz_add", methods={"GET|POST"}) */  
}
```

## D - Communication entre nos 3 composants

Comme nous venons de le voir, chacun de ses composants a sa fonction, mais voyons un peu comment ils opèrent en synergie pour fournir à l'utilisateur, une réponse à sa requête :

1. Commençons par le début. Comment l'utilisateur contact notre application ?

Dans le cadre de notre application web sous Symfony, il commence par entrer une url dans son navigateur. Il contacte donc une Route.

En fonction de la Route empruntée par notre utilisateur, c'est notre Contrôleur qui va déterminer quelles sont les actions à entreprendre.

2. Quelles données doivent être renvoyées ? À ce moment-là, notre Contrôleur va demander au Modèle (en passant par l'ORM Doctrine) les données.

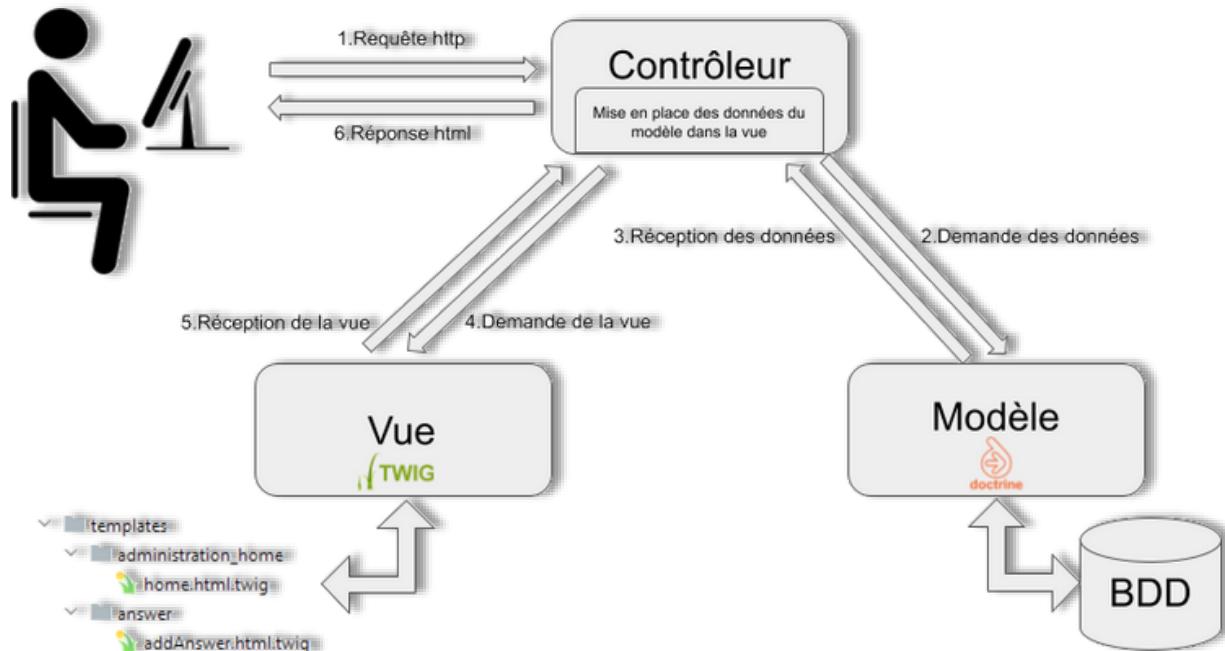
3. Le Modèle va donc renvoyer à notre Contrôleur, les données requêtées.

4. Comment doivent-elles être présentées ? Dans le même temps, le Contrôleur détermine quelle Vue doit être utilisée dans sa réponse, il va donc la rechercher dans les templates (Twig)

5. Le template et les données récupérés, le Contrôleur se charge de mettre en forme les données dans la vue

6. Sa réponse http étant prête, le Contrôleur envoie sa réponse à l'utilisateur.

Schéma représentant le design pattern MVC



## E – L'architecture 3 tiers

Maintenant que nous avons examiné en détail le pattern MVC et compris comment il permet de structurer notre application en séparant les responsabilités de manière efficace, il est temps d'élargir notre vision et d'explorer une architecture plus globale : l'architecture 3 tiers.

L'architecture 3 tiers est un modèle d'organisation logique qui vise à diviser une application en trois couches distinctes : la présentation, la logique métier et les données. Chaque couche est responsable de tâches spécifiques et communique avec les autres couches selon des règles bien définies.

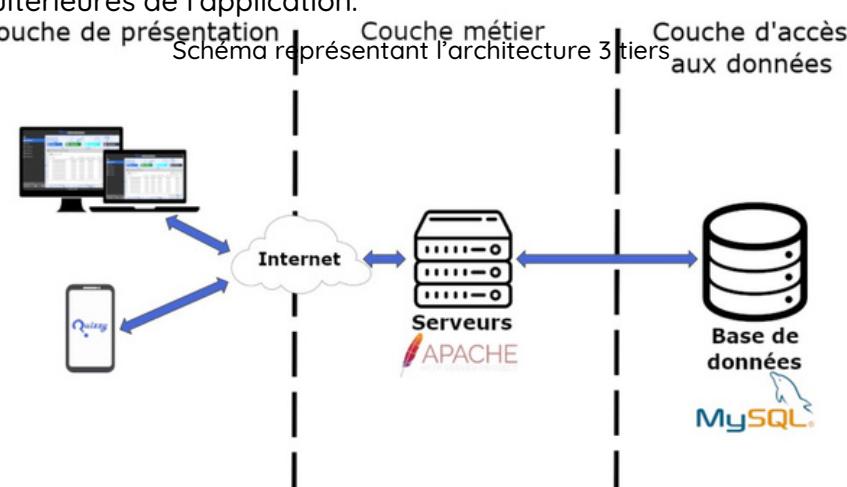
La première couche, la présentation ou couche d'interface utilisateur, est chargée de l'interaction avec les utilisateurs finaux. Elle comprend l'interface graphique, les contrôles et les éléments visuels qui permettent aux utilisateurs d'interagir avec l'application. Cette couche est souvent implémentée en utilisant le pattern MVC, où le contrôleur gère les interactions utilisateur et coordonne les actions à effectuer.

La deuxième couche, la logique métier ou couche de traitement, contient la logique métier de l'application. C'est ici que les règles métier, les calculs complexes et les opérations spécifiques à l'application sont mises en œuvre. Cette couche utilise les données provenant de la couche de données et les transforme en résultats significatifs pour les utilisateurs.

Enfin, la troisième couche, la couche d'accès aux données, est responsable de l'accès et de la gestion des données de l'application. Elle peut être liée à une base de données, à des services web ou à d'autres sources de données. Cette couche fournit les

mécanismes nécessaires pour récupérer, stocker et manipuler les données de manière sécurisée. En adoptant une architecture 3 tiers, nous pouvons bénéficier d'une meilleure

modularité, d'une séparation claire des responsabilités et d'une évolutivité accrue. Chaque couche peut être développée et testée indépendamment, ce qui facilite la maintenance et les mises à jour ultérieures de l'application.



# X - Sécurité

Dans le paysage numérique actuel, la sécurité des applications web est devenue une préoccupation majeure. Les applications web jouent un rôle essentiel dans la communication, les transactions en ligne et le stockage des données sensibles. Cependant, elles sont également vulnérables aux attaques malveillantes et aux violations de la confidentialité. Les cybercriminels utilisent des techniques sophistiquées pour exploiter les failles de sécurité des applications web et accéder aux données sensibles des utilisateurs. Les conséquences peuvent être désastreuses pour les entreprises et les utilisateurs, allant de la perte de données à la violation de la vie privée.

La protection des utilisateurs, la préservation de l'intégrité des données et la

disponibilité

des services sont des aspects essentiels pour assurer la confiance et la fiabilité d'une application web. C'est pourquoi la sécurité est un enjeu crucial à prendre en compte lors du développement d'une application web. Les développeurs doivent être conscients des risques liés à la sécurité et mettre en place des mesures de sécurité efficaces pour protéger les utilisateurs et les données sensibles.

A - Les attaques XSS (Cross-Site Scripting)

Une attaque XSS est une technique courante pour injecter du code malveillant dans des sites web et exploiter les vulnérabilités des navigateurs des utilisateurs. L'attaque XSS peut se produire lorsqu'une application web n'effectue pas une validation ou un échappement approprié des données entrées par les utilisateurs.

Dans le contexte d'une application en PHP, une attaque XSS peut se produire si des données fournies par l'utilisateur sont affichées directement dans les pages web sans être correctement filtrées ou échappées. Par exemple, si un utilisateur malveillant entre du code JavaScript dans un champ de saisie et que cette entrée est affichée sans protection dans une page web, le code JavaScript sera exécuté par le navigateur de l'utilisateur, ce qui peut entraîner des actions indésirables telles que le vol de données sensibles, la modification de contenu ou la redirection vers des sites malveillants.

Pour prévenir les attaques XSS, il est essentiel de mettre en place des pratiques de sécurité appropriées. Dans le cas d'une application Symfony, cela peut être réalisé en utilisant les mécanismes de sécurité intégrés du Framework, tels que l'échappement automatique des données lors de l'affichage dans les vues à l'aide de balises spécifiques (« `htmlSpecialChars` » en PHP ou « `| escape` », un équivalent pour Twig), ou en utilisant

des

composants de sécurité supplémentaires tels que le composant Security pour gérer l'authentification et l'autorisation.

## B - Les injections SQL

Une attaque par injection SQL est une technique courante utilisée par les pirates informatiques pour exploiter les vulnérabilités d'une application web et accéder à une base de données. L'attaque consiste à injecter du code SQL malveillant dans une requête SQL de l'application, afin de manipuler le comportement de la base de données et obtenir des informations sensibles ou effectuer des opérations non autorisées.

Dans le contexte de cette application développée sous le Framework Symfony, il est important de comprendre comment une telle attaque peut se produire et comment s'en prémunir. Dans Symfony, les requêtes SQL sont générées à l'aide de l'ORM Doctrine.

Cela

signifie que les requêtes SQL sont générées automatiquement à partir des interactions avec les objets de la base de données.

L'attaque par injection SQL peut se produire si l'application ne filtre pas correctement

les

entrées utilisateur avant de les utiliser dans les requêtes SQL. Par exemple, si un utilisateur est autorisé à entrer du texte dans un formulaire de recherche et que cette entrée n'est pas correctement validée ou nettoyée, un attaquant pourrait saisir du code SQL malveillant qui sera exécuté directement dans la requête SQL.

Le résultat de cette attaque peut être dévastateur. L'attaquant peut extraire, modifier

ou

supprimer des données dans la base de données, voire prendre le contrôle total du système. Cela peut entraîner la divulgation d'informations confidentielles, la perte de données ou des atteintes à la vie privée des utilisateurs.

Pour se prémunir contre les attaques par injection SQL, il est essentiel de mettre en

place

des mesures de sécurité appropriées. Dans une application Symfony, cela peut être réalisé en utilisant les fonctionnalités de sécurité intégrées du Framework, telles que les requêtes préparées, qui garantissent que les entrées utilisateur sont correctement échappées et ne peuvent pas être interprétées comme du code SQL.

```
1 usage  ↗ Bruno BARON
public function findStaff(): array
{
    $query = $this->createQueryBuilder('user')
        ->where('user.roles LIKE :role')
        ->setParameter('role', '%ROLE_ADMIN%')
        ->orWhere('user.roles LIKE :role2')
        ->setParameter('role2', '%ROLE_MODERATOR%')
        ->andWhere('user.deletedAt IS NULL')
        ->getQuery()
        ->getResult();
}
```

Exemple de requête préparée

## C - Les attaques CSRF (Cross-Site Request Forgery)

Une attaque CSRF est une technique d'attaque courante qui exploite la confiance que les applications web accordent aux utilisateurs authentifiés. L'attaque CSRF se produit lorsqu'un attaquant réussit à tromper un utilisateur authentifié pour qu'il effectue involontairement des actions indésirables sur une application sans son consentement.

Dans le contexte d'une application sous le Framework Symfony, une attaque CSRF peut se produire lorsque des actions sensibles sont effectuées via des formulaires ou des liens.

L'attaque se déroule en plusieurs étapes :

- L'attaquant crée un site malveillant contenant un formulaire ou un lien qui effectue une action spécifique sur l'application ciblée. Par exemple, il peut s'agir de la modification du mot de passe d'un utilisateur ou de la suppression d'un élément important.
- L'attaquant incite l'utilisateur authentifié à visiter son site malveillant. Cela peut se faire par le biais de techniques telles que l'envoi de liens trompeurs par e-mail, les messages sur les réseaux sociaux, ou l'inclusion du lien dans des sites web compromis.
  - L'utilisateur authentifié, étant déjà connecté à l'application cible, visite le site malveillant. Sans se méfier, il soumet involontairement le formulaire ou clique sur le lien malveillant.
  - Le navigateur de l'utilisateur envoie la requête contenant l'action indésirable à l'application cible, en incluant les informations d'authentification du compte de l'utilisateur. Étant donné que l'application fait confiance à l'utilisateur authentifié, elle traite la requête et effectue l'action demandée, pensant qu'elle provient de l'utilisateur lui-même.

Pour se prémunir contre les attaques CSRF, Symfony propose des mécanismes de protection intégrés. L'un de ces mécanismes est l'utilisation de tokens CSRF qui sont générés et associés aux formulaires dans l'application. Ces tokens sont ensuite vérifiés lors de la soumission du formulaire pour s'assurer que la requête provient bien du site lui-même et non d'une source externe.

Au travers de l'application Quizzy, et dans les application sous le Framework Symfony

en

général, on activera ce mécanisme dans le fichier de configuration « framework.yaml » situé dans le répertoire « config/packages » contenant tous les fichier de configuration des packages utilisés dans l'application.

Activation de la sécurisation des formulaires par token CSRF dans Symfony

```
framework:  
    secret: '%env(APP_SECRET)%'  
    csrf_protection: true  
    http_method_override: false
```

En utilisant la fonctionnalité de protection CSRF de Symfony, chaque formulaire générera automatiquement un token CSRF qui sera vérifié lors de la soumission du formulaire. Cela garantit que seules les requêtes provenant du site lui-même seront acceptées, tandis que les requêtes externes, telles que celles provenant d'un site malveillant, seront rejetées. Dans l'exemple ci-dessous, on peut voir que dans la page générée dans le navigateur, Symfony a bien procédé à l'ajout d'un « input » caché contenant comme valeur le token CSRF :

Extrait d'un formulaire généré par l'application dans un navigateur Web

```
▼ <form name="question_base" method="post">
  ▶ <div class="mb-3"> ... </div>
  ▶ <div class="mb-3"> ... </div>
  ▶ <div class="mb-3"> ... </div>
  ▶ <fieldset class="mb-3"> ... </fieldset>
  ▶ <div class="d-none"> ... </div>
  ▶ <div class="d-flex flex-column flex-md-row mt-5 align-items-center justify-content-md-end"> ... </div> [flex]
    <input id="question_base_token" type="hidden" name="question_base[_token]" value="7b0d75db36.Cat_bRfwE1aqnGfzRobswGhdN1SRh2-
      bRaGxDWCNgmI.e55GFWazajmZ5CiVG8ujji-4zzByS7oDOlNsWbs1jZBnTdaQIxMe-fsvw">
</form>
```

input contenant le token CSRF

En conclusion, une attaque CSRF est une tentative de tromper un utilisateur authentifié pour qu'il effectue involontairement des actions indésirables sur une application web. Symfony propose des mécanismes de protection, tels que les tokens CSRF, pour prévenir efficacement les attaques CSRF et garantir la sécurité des applications web développées avec le Framework.

# XI - Politique de test

Le processus de développement d'un logiciel est complexe et comporte de nombreux défis. L'un de ces défis majeurs est de s'assurer que le logiciel développé fonctionne correctement et répond aux exigences spécifiées. Pour garantir la qualité et la fiabilité du logiciel, il est essentiel de mettre en place des tests appropriés.

Ici, nous aborderons les tests unitaires et les tests d'intégration, qui sont deux types de tests essentiels dans le domaine du développement logiciel. Les tests unitaires se concentrent sur la vérification du bon fonctionnement de chaque composant logiciel de manière isolée, tandis que les tests d'intégration examinent les interactions entre les différents composants pour s'assurer qu'ils fonctionnent harmonieusement ensemble.

Nous allons explorer les principes fondamentaux des tests unitaires et d'intégration,

ainsi

que les bonnes pratiques associées à leur mise en œuvre.

## A - Les tests unitaires

Les tests unitaires sont une méthode de test qui permet de vérifier le bon fonctionnement d'une partie précise d'un logiciel ou d'une portion d'un programme (appelée « unité » ou « module »). Dans ce chapitre, nous allons nous concentrer sur les tests unitaires pour le login.

Nous allons présenter deux tests : un test de succès et un test d'échec. Le test de succès vérifie que l'utilisateur peut se connecter avec des identifiants valides. Le test d'échec vérifie que l'utilisateur ne peut pas se connecter avec des identifiants invalides.

Les tests unitaires sont une étape importante dans le développement de logiciels car ils permettent de détecter les erreurs dès le début du processus de développement. Cela permet de réduire les coûts et les délais de développement en évitant les erreurs coûteuses à corriger plus tard.

Pour la réalisation de ses tests, une base de données de test a été mise en place :

Extrait de la base de données de test

The screenshot shows a MySQL database interface with the following details:

- Server:** 127.0.0.1
- Base de données:** quizzy\_test
- Table:** user

The table structure is as follows:

	id	email	roles (DC2Type:json)	password	username	date_c
1	1	user@test.com	["ROLE_USER"]	\$2y\$10\$Wp0c73wg9JEMEnVbIL3uL0mW0yV8nJaYQBNTXM/ual...	user	2003-0
2	2	admin@test.com	["ROLE_ADMIN"]	\$2y\$04\$AJF9P302JRuP0FYKDL6ND.TEA.OHE2HhNM8oMr7cUMm...	admin	2003-0
3	3	moderator@test.com	["ROLE_MODERATOR"]	\$2y\$04\$2S9NGQj0yA5w3uR2USUE8.fNl6cQU9emdO1g3VqjiUb...	moderator	2003-0

A tooltip is displayed over the table header, stating: "Utilisateur fictifs, créés dans le but de servir à la réalisation des tests".

Extrait du fichier env.test, contenant les variable d'environnement pour les test dont l'adresse de la base de données de test

```
# define your env variables for the test env here
KERNEL_CLASS='App\Kernel'
APP_SECRET='$secretf0rt3st'
SYMFONY_DEPRECATED_HELPER=999999
PANTHER_APP_ENV=panther
PANTHER_ERROR_SCREENSHOT_DIR=/var/error-screenshots

DATABASE_URL="mysql://root:@127.0.0.1:3306/quizzz"
```

### Test d'échec de connexion

± Bruno BARON

```
public function testLoginUnknownUser(): void
{
    $client = static::createClient();

    $crawler = $client->request( method: 'GET', uri: '/login');

    $form = $crawler->selectButton( value: 'Se connecter')->form([
        'email' => 'unknown@test.com',
        'password' => 'password'
    ]);
    $client->submit($form);
    $this->assertResponseRedirects( expectedLocation: '/login');
    $client->followRedirect();
    $this->assertSelectorExists( selector: '.alert.alert-danger');
}
```

Nom de fonction illustrant bien ce qui est testé, ici, une tentative de connexion par un utilisateur inconnu

Ici, on accède à la page de login

on complète le formulaire de connexion et on le soumet

on vérifie que l'utilisateur est bien redirigé vers la page de login et qu'un message d'erreur s'affiche

### Test de connexion réussie

± Bruno BARON

```
public function testLoginModeratorRole(): void
{
    $client = static::createClient();

    $crawler = $client->request( method: 'GET', uri: '/login');

    $form = $crawler->selectButton( value: 'Se connecter')->form([
        'email' => 'moderator@test.com',
        'password' => 'password'
    ]);
    $client->submit($form);
    $this->assertResponseRedirects( expectedLocation: '/');
    $this->assertSelectorNotExists( selector: '.alert.alert-danger');
}
```

ici, on est dans un cas de succès de connexion, on doit être redirigé vers la page home, et aucune alerte ne doit être affichée

## B - Les tests d'intégration

Les tests d'intégration sont une méthode de test qui permet de vérifier que les différentes parties d'un système fonctionnent correctement ensemble. Dans ce chapitre, nous allons nous concentrer sur les tests d'intégration pour la création d'un quiz.

Nous allons présenté un scénario de test pour la création d'un quiz qui vérifie que les champs obligatoires sont bien remplis et que l'utilisateur est redirigé vers la bonne page après avoir validé le formulaire.

Ces scénarios de test peuvent être effectués manuellement ou automatisés à l'aide d'outils de test. Ici, nous les avons réalisé manuellement dans un premier temps, et nous les avons consigné dans des fichiers .txt

Test d'intégration de la fonction de création de quiz

```
AddQuizTest.txt ×
1 1- Log in to your account as an administrator or moderator.
2 2- Go to the quiz creation page (/quizzes/add).
3 3- Fill in the required fields: name, description and status (private or public).
4 4- Fill in the optional field: category.
5 5- Click on the "Validate" button.
6 6- Check that you are redirected to the creation page of your first question.
7 7- Check that the quiz is created and that its values are the correct ones in the database.
```

Les tests d'intégration sont une étape importante dans le développement de logiciels car ils permettent de détecter les erreurs dès que les différentes parties du système sont intégrées. Cela permet de réduire les coûts et les délais de développement en évitant les erreurs coûteuses à corriger plus tard.

# XII - Veille Technologique

## A - Veille globale

La veille informatique est une pratique importante pour rester informé des dernières évolutions technologiques et des bonnes pratiques de son domaine. Pour faire de la veille informatique, il faut identifier ses objectifs, ses besoins, ses sources et ses outils de veille. On peut utiliser des flux RSS, Twitter, Google Alert, des newsletters, des conférences ou des discussions avec ses pairs.

Pour ma part, moi qui suis passionné par les nouvelles technologies et constamment à l'affût des dernières tendances et innovations dans ce domaine, je suis convaincu que la veille technologique est essentielle pour rester compétitif dans ce secteur en constante évolution.

Pour cela, je pense que la curation de contenu est un élément clé de la veille technologique. En effet, il est important de sélectionner les sources d'information les pertinentes et de filtrer les informations pour ne retenir que celles qui sont les plus utiles. Pour cela, j'utilise des sites tels que <https://compagnon.artisandev.com/veille> et <https://feedly.com> pour suivre les dernières actualités technologiques.

<https://compagnon.artisandev.com/veille>

The screenshot shows a news item from the 'Quoi de neuf les devs ?' newsletter, issue #34. The article features a cartoon rabbit in a suit pointing towards a circular portrait of a man with a beard. The text below the image reads: 'Un concentré d'infos pour les devs par un dev. Cette semaine avec l'interview de Steve Mc Dougall alias @JustSteveKing'. A green button at the bottom says 'Afficher la ressource'.

J'ai également configuré un flux d'actualité Google sur mon mobile pour rester informé en déplacement.

Bien que je ne participe pas régulièrement à des conférences ou des événements, je suis intéressé à en apprendre davantage sur les dernières tendances et innovations dans le domaine des nouvelles technologies. Je communique également avec mes collègues pour rester informé des dernières tendances et innovations dans ce domaine.

## B – Veille sécurité

Parmi les différents aspects de la veille technologique, il faut indéniablement compter sur la sécurité car il s'agit d'un enjeu majeur, en effet, elle concerne la protection des données, des systèmes et des utilisateurs face aux menaces et aux risques liés au cyberspace.

Pour réaliser une veille efficace sur la sécurité, il est nécessaire de consulter des sources fiables et reconnues, qui fournissent des informations pertinentes, actualisées et adaptées aux besoins des professionnels. Parmi ces sources, deux sites se distinguent par leur qualité et leur notoriété : le site de l'Agence nationale de la sécurité des systèmes d'information (ANSSI) et le site de l'Open Web Application Security Project (OWASP).

Le site de l'ANSSI est le portail officiel de la cybersécurité en France. Il propose des actualités, des guides, des recommandations, des outils et des services pour aider les acteurs publics et privés à se protéger contre les cyberattaques. Le site de l'ANSSI couvre tous les aspects de la sécurité des systèmes d'information, tels que la gouvernance, la gestion des risques, la défense en profondeur, la sensibilisation, la réaction aux incidents, etc.

<https://www.ssi.gouv.fr>

The screenshot shows the ANSSI website's navigation bar with links for social media, declaration of vulnerability, emergency cases, alerts, press, recruitment, and language selection. Below the header, there are tabs for 'PRINCIPALES MENACES', 'PRÉCAUTIONS ÉLÉMENTAIRES', 'BONNES PRATIQUES', 'LOGICIELS PRÉCONISÉS PAR L'ANSSI', 'FORMATIONS', and 'GLOSSAIRE'. A sub-menu for 'MÉTHODOLOGIE' is selected. The main content area features a heading 'BONNES PRATIQUES' and a section titled 'Méthodologie' with three downloadable PDF files:

TITRE	THEME	DATE	PDF
LES MESURES CYBER PRÉVENTIVES PRIORITAIRES	Méthodologie	17/05/2023	652.96 Ko
ORGANISER UN EXERCICE DE GESTION DE CRISE CYBER	Méthodologie	14/10/2020	1.88 Mo
MAÎTRISE DU RISQUE NUMÉRIQUE - L'ATOUT CONFIANCE	Méthodologie	18/11/2019	909.21 Ko

To the right, a 'Nuage de tags' displays various cybersecurity terms such as administration, architecture, authentication, chiffrement, cloisonnement, commutateur, conteneur, critères de sécurité, cryptographie, défense en profondeur, firewall, formation, Génos, homologation, ICS, interconnection, Internet, IPSEC, journalisation, méthodologie, nomalisme, NTP, pare-feu, passeur, passerelle, prévention, protection, PSSI, PSIE, qualification, règle, réglementation, réseau, RGS, serveur, stratégie, supports amovibles, système industriel, systèmes industriels, TLS, virtualisation, VPN, Windows, Windows 10.

Le site de l'OWASP est une communauté internationale qui promeut la sécurité des applications web. Il offre des ressources gratuites et ouvertes, telles que des standards, des méthodologies, des projets, des événements et des formations pour aider les développeurs, les testeurs, les auditeurs et les utilisateurs à concevoir, développer, tester et maintenir des applications web sécurisées. Le site de l'OWASP est notamment connu pour son Top 10 des risques de sécurité applicative, qui est une référence mondiale pour identifier et prévenir les vulnérabilités les plus courantes et les plus critiques.

<https://owasp.org>

The screenshot shows the OWASP website's header with links for 'PROJECTS', 'CHAPTERS', 'EVENTS', 'ABOUT', 'Member Login', 'Store', 'Donate', and 'Join'. Below the header, there is information about a 'Meetup OWASP Paris (17/04/2023)' at Theodo. The 'Lightning talks' section lists several items:

- Bearer: discovery sensitive data
  - Utilise Tree Sitter
  - Une vulnérabilité connue (cache configuration change leading to account vulnerability) a pu être retrouvée sur NodeGoat
- Disparition des mots de passe: Passwordless
  - WebAuthN: Yubikey, Smartphone, TouchId, ...
  - Sync de vos clés secrètes entre vos équipements avec Passkey
- Un side project d'application Medical en intégrant la sécurité
- 2Captcha: l'outil de bypass de captcha supporte désormais Cloudflare Turnstile

The 'Workshop' section lists:

- Démonstration d'une pipeline sécurité avec des outils Open Source (Semgrep, Trivy, ...) et un reporting centralisé avec Defect Dojo
- Mise en place d'une authentification WebAuthN avec SimpleWebAuthn dans une application NestJS

The 'Upcoming OWASP Global Events' section lists:

- OWASP Global AppSec Singapore 2023
- October 4-5, 2023
- OWASP Global AppSec Washington DC 2023
- October 30 - November 3, 2023
- OWASP Global AppSec San Francisco 2024
- September 23-27, 2024
- OWASP Global AppSec Washington DC 2025

## XIII - Difficultés Rencontrées

La mise en place d'ApiPlatform sur mon application Symfony a été l'une des tâches les plus difficiles que j'ai rencontrées au cours de ce projet. Bien que j'ai déjà travaillé avec Symfony, je n'avais jamais utilisé ApiPlatform auparavant et j'ai rapidement réalisé que cela allait être un défi.

L'un des principaux problèmes que j'ai rencontrés était le manque de documentation claire et concise sur la façon d'utiliser ApiPlatform avec Symfony. Bien qu'il y ait beaucoup de ressources en ligne, il était difficile de trouver des informations précises et à jour sur la façon de configurer et d'utiliser ApiPlatform. J'ai toutefois trouvé la solution à mon problème grâce un lien vers le site « stackoverflow » depuis le site même d'ApiPlatform.

J'ai également rencontré des problèmes avec la configuration des tokens JWT, ce qui représentait un blocage inacceptable si je souhaitais fournir une application un minimum sécurisée.

Malgré ces difficultés, j'ai travaillé dur pour surmonter ces obstacles et mettre en place ApiPlatform sur mon application Symfony. J'ai passé beaucoup de temps à lire la documentation et à chercher des solutions en ligne, et j'ai finalement réussi à mettre en place la configuration requise pour répondre aux besoins de mon application.

## XIV - Conclusion

Mon projet de développement d'une application web de quiz a été une expérience incroyablement enrichissante. J'ai utilisé les technologies Symfony et Angular pour créer une application web qui se veut robuste et conviviale et qui répond aux besoins de mes utilisateurs.

Au cours du projet Quizzy, j'ai rencontré plusieurs défis techniques, mais j'ai surmonté les difficultés et créé une application web qui est à la fois fonctionnelle et, je l'espère, élégante.

Quizzy m'aura beaucoup apporté, techniquement parlant, j'ai énormément appris, notamment la création d'une API, mais aussi d'un point de vue méthodologique, car on se rend compte que si on manque juste un peu de rigueur, on peut très vite prendre énormément de retard.

Je suis fier du travail que j'ai accompli et je suis convaincu que mon application web de quiz pourrait être un outil de détente précieux pour mes utilisateurs. Pour la suite, je ne compte pas laisser Quizzy de côté, je pense bien continuer à améliorer ce site et lui apporter autant de fonctionnalité qu'on peut en imaginer pour ce genre d'application.



## XV – Remerciements



## Annexe

Cahier des charges du projet :



# Sommaire

Sommaire.....	2
<b>Présentation générale.....</b>	<b>3</b>
<b>Objectifs.....</b>	<b>3</b>
<b>Cibles &amp; Caractéristiques.....</b>	<b>3</b>
<b>Description du contenu.....</b>	<b>4</b>
<b>Planning.....</b>	<b>6</b>
<b>Exigences Fonctionnelles.....</b>	<b>6</b>
I. Principales composantes.....	7
II. Fonctionnalités "Front Office" .....	7
III. Fonctionnalités "Back Office" : Administration.....	8
IV. Confidentialité.....	9
V. Droits d'accès.....	10
VI. Authentification.....	10
Exigences Technique.....	10
Charte	
Graphique..Logo.....	11
11	
II.	
Couleurs	
et	
fonts.....	11
III.	
Minimum Background.....	11
12	Délai
Environnement de développement.....	12.... 13

## Présentation générale

Quizzy a pour but d'être un site web innovant qui propose des quiz interactifs pour divertir et informer les utilisateurs de tous âges.

Notre site web est conçu pour offrir une expérience utilisateur intuitive et captivante, avec des quiz amusants et éducatifs sur des sujets variés tels que la culture générale, l'histoire, la géographie, la musique, le cinéma et bien plus encore.

Que vous soyez à la recherche d'un moyen de vous détendre et de vous amuser, ou, que vous souhaitiez tester vos connaissances sur un sujet en particulier, Quizzy sera la solution idéale pour vous.

## Objectifs

1. **Divertir tous les visiteurs du site quelque soit leur âge**
2. **Offrir un large choix de quiz entièrement catégorisé**
3. **Proposer différents formats de questions (QCM texte ou image, réponse libre, etc ...)**
4. **Permettre la création de quiz ou de questions à tout utilisateur enregistré**
5. **Proposer un mode multijoueur**
6. **Gérer sa liste d'amis**
7. **Présenter un tableau des meilleurs scores pour chaque quiz ainsi que les statistiques du joueur**

## Cibles & Caractéristiques

1. **Coeur de cible** : Toute la population située grossièrement entre 20 et 40 ans
2. **Cibles secondaire** : Le reste de la population, et sur le long terme, d'autres territoires que la France

## Description du contenu

**La page d'accueil** : ici, si l'utilisateur n'est pas authentifié, un premier conteneur affichera une présentation globale du site. Cette présentation sera suivie par 2 liens : "inscription" et "connexion".

Après identification de l'utilisateur, la page affichera une liste des 10 derniers quizz ajoutés, suivie d'une liste de 10 quizz aléatoires elle-même suivie d'un lien vers la page "Quizzs".

À la suite de ces quizzs, une liste 10 thèmes aléatoires seront suggérés à l'utilisateurs, cette liste sera suivie d'un lien vers la page des "Thèmes"

**La page Thèmes** : cette page proposera, comme son nom l'indique, la liste complète et paginée des thèmes proposés sous forme de "card" cliquable et redirigeant vers la page du thème sélectionné. On y trouvera aussi une barre de recherche afin d'affiner la sélection affichée sur la page.

**Les pages "Thème"** : ces pages présenteront le thème sélectionné grâce à sa description complète ainsi que, le cas échéant, les sous-thèmes associés. Une liste aléatoire de quizzs (nombre à déterminer) de ce thème, ou de ses sous-thèmes sera proposée. Cette liste sera

bien

sûr suivie d'un lien vers la page "Quizzs" où un filtrage sur ce thème sera appliqué.

**La page "Quizzs"** : comme son homologue "Thèmes", cette page affichera la liste complète et paginée de l'intégralité des quizzs proposés sur le site (par défaut, dans l'ordre du plus récent au plus ancien). Chaque quiz sera présenté avec son thème associé, son nombre de questions, et son niveau de difficulté ainsi que le dernier meilleur score enregistré.

Cette page disposera aussi de sa barre de recherche spécifique, néanmoins celle-ci sera un

peu

plus détaillée, il sera en effet possible de filtrer les quizzs par thème, difficulté, ou encore le nombre de questions, de même, il sera aussi possible d'ordonner l'affichage par ordre croissant ou décroissant de leur, date de création, difficulté ou encore nombre de questions.

**Les pages “Quiz” :** dans un premier temps, elles n'afficheront qu'un tableau des 10 meilleurs scores réalisés sur le quiz choisi et un texte indiquant de cliquer sur le bouton qui suivra pour démarrer le quiz. À terme, un second bouton sera ajouté pour lancer le quiz en multijoueur. Une fois le quiz lancé, les questions s'afficheront une par une, soit après écoulement d'un compte à rebours, soit après que le joueur ait validé sa réponse. En fin de quiz, il est bien entendu que le joueur aura accès à son score mais aussi aux bonnes réponses. S'il s'avérait qu'une question ait eu une formulation trompeuse et/ou erronée, ou bien que la bonne réponse affiché ne soit pas réellement la bonne, il sera évidemment possible pour notre utilisateur d'effectuer un signalement de la question au travers d'un petit formulaire l'invitant à indiquer le/les problème(s) relevé(s). S'il y avait trop de questions à signaler, il serait aussi possible d'effectuer un signalement du quiz complet.

**Les pages “Création” :** comme leur nom l'indique, elles serviront à la création, création de quiz ou de question unique. En effet, l'utilisateur aura l'entièrre liberté de créer ce type de contenu et de le proposer à tous, ou alors de le garder pour lui et son entourage. Il est à préciser que si notre créateur opte pour une publication de son contenu, celui-ci sera soumis à la modération qui s'assurera que la proposition ne contient pas d'erreur bien sûr mais aussi que rien dans cette publication ne relèvera d'un caractère pouvant choquer ou heurter la sensibilité du public.

# Planning

## ● Phase 1 - Conception

- Conception d'une première ébauche du modèle de données - Zoning, Wireframe & Maquette
- Choix et acquisition du nom de domaine

## ● Phase 2 - Développement

- Initialisation du projet Symfony, création de la base de données de dev, et développement des premières fonctions de base
- Initialisation de l'API qui permettra le lien entre le Back End et le Front End
- Initialisation du Front End Angular, et création des premières pages du site
- Développement des fonctions plus spécifiques, nécessaire à la suite du développement du Front End
- Finalisation du développement Front End

## ● Phase 3 - Test

- Utilisation du site avec les quizzes présents en base de donnée et contrôle du bon affichage des résultats
- Test de création de questions et quizzes complets, et vérification de la bonne soumission à la modération

## ● Phase 4 - Déploiement

- Création de la base de donnée pour l'environnement de production - Déploiement du projet sur l'hébergement web acquis en phase 1
- Test de toutes les fonctionnalités du site "à vide"

# Exigences Fonctionnelles

## I. Principales composantes

Le site devra pouvoir permettre à un visiteur de créer un compte afin qu'il puisse participer à un quiz.

## II. Fonctionnalités "FrontOffice"

Le site doit permettre de pouvoir accéder aux fonctionnalités suivantes :

- Paged'accueil : elle contiendra l'ensemble des éléments de navigation ainsi que des liens directs vers des quiz choisis aléatoirement à chaque rafraîchissement de pages.
  - Un menu de navigation : il permettra aux utilisateurs de naviguer entre les différentes rubriques du site.
- Une page "Mon compte" : elle devra permettre à l'utilisateur d'accéder à toutes les informations enregistrées sur le site et de les modifier si besoin. Il devra pouvoir en outre, supprimer son compte s'il le souhaite.
- Une page "Quizzes" : Elle affichera 3 thèmes choisis aléatoirement, et sous chacun d'eux 5 quizzes eux-mêmes choisis aléatoirement. Ce contenu sera suivi d'un lien vers la page recensant tous les thèmes présents sur le site.
- Une page "Mes Quiz et Questions" : sur cette page, il sera possible pour l'utilisateur d'accéder à un outil de création de quiz ou à un outil de création de question. De même, il pourra accéder à la liste des quizzes et questions qu'il aura pu créer et aura la possibilité de les corriger/mettre à jour selon son gré.
- Une page "Contact" : l'utilisateur trouvera un formulaire qui lui permettra de contacter l'administration du site afin de poser des questions, remonter d'éventuels dysfonctionnements, ou encore proposer des axes d'améliorations possibles de l'application
- Une page "Plan du site"
- Une page "Mentions légales"

### **III. Fonctionnalités "BackOffice":Administration**

L'administration (voir plus bas), sera accessible par un login/mot de passe attribué à une personne physique.

**L'interface sera accessible via navigateur internet.**

Il est important de définir au moins 3 profils et leurs droits :

- **Modérateur:** leur rôle étant de veiller à ce qu'aucun contenu à caractère echoquant ne puisse être rendu visible au public mineur ou sensible du site, ces derniers devront procéder à un contrôle de chaque nouveau quiz ou question proposé avec un statut public, de plus, des questions signalées comme ayant des formulation ou réponses erronées devront pouvoir être mise à jour. Toutefois, bien qu'ils puissent masquer au public un contenu, il n'est pas question que les modérateurs puissent en supprimer :
  - ajout / modification des quizs "publics"
  - ajout / modification des questions "publics"
  - ajout / modification des thèmes
  - accès aux mails reçus de la part des administrateurs

- **Administrateur :** gestionnaire du contenu proposé sur le site, il aura un accès quasi total aux fonctionnalités d'administration du site :
  - ajout / modification / suppression des profils Modérateur
  - ajout ou suppression de compte utilisateur (seulement sur demande particulière en cas de dysfonctionnement)
  - ajout / modification / suppression des quizs "publics"
  - ajout / modification / suppression des questions "publics"
  - ajout / modification / suppression des thèmes
  - accès aux mails reçu via le formulaire de contact et répartition de ces derniers aux Administrateurs ou Modérateurs en fonction de l'action ou réponse souhaitée

- Super Administrateur : Webmaster et gestionnaire du site, il aura un **accès total** à toutes les fonctionnalités d'administration du site :
  - ajout / modification / suppression des profils Administrateur et Modérateur
  - suppression de compte utilisateur (seulement sur demande particulière en cas de dysfonctionnement)
  - ajout / modification / suppression des quizz
  - ajout / modification / suppression des questions
  - ajout / modification / suppression des thèmes
  - accès aux mails reçu via le formulaire de contact et répartition de ces derniers aux Administrateurs ou Modérateurs en fonction de l'action ou réponse souhaitée
- Utilisateur (public) : une fois son compte créé, l'utilisateur doit bien sûr pouvoir accéder au contenu du site mais aussi accéder à la gestion de ses informations personnelles, il doit aussi avoir accès aux outils de création de quiz ou question ainsi qu'à la gestion du contenu de sa création, un mode multijoueur devant aussi être disponible, il devra aussi être possible de gérer une liste d'amis :
  - consultation / modification / suppression de son compte
  - ajout de quizz
  - consultation / modification / suppression de ses quizz "privés"
  - ajout de questions
  - consultation / modification / suppression de ses questions "privés"
  - ajout / consultation / suppression d'autres utilisateur dans sa liste d'amis

#### IV. Confidentialité

Il s'agira de protéger l'accès en lecture ou en écriture aux informations "privées" de chaque utilisateur, toutefois, il faudra pouvoir supprimer un compte si la demande en était faite par l'utilisateur (en cas de dysfonctionnement par exemple).

De même, les quizz ou questions créés par un utilisateur avec un statut privé ne devront en aucun cas être visibles ou modifiables pour l'administration du site ou d'autres utilisateurs ne faisant pas partie de la liste d'amis du créateur du contenu en question.

## V. Droits d'accès

L'accès sera contrôlé selon le profil d'utilisateur. Pour chaque type d'information, les droits d'accès (lecture, écriture) devront être déterminés par la catégorie et par l'identité de l'utilisateur.

## VI. Authentification

Le site devra procéder à une identification et une authentification sécurisées des utilisateurs afin d'assurer la traçabilité de leurs actions, ainsi que l'intégrité et la non falsification des informations.

# Exigences Technique

Le site devra pouvoir être accessible depuis la plupart des navigateurs actuels et à minima les 3 suivants :

- Chrome
- Firefox
- Edge

Concernant les langages de programmation, le backend et le frontend destinée à l'administration seront développé sous le framework Symfony (PHP, et twig pour l'interface administrateur)

Pour ce qui est de l'interface utilisateur, cette dernière sera développée avec le framework Angular (typescript)

Enfin, pour ce qui est de la base de données, celle-ci sera hébergée sur un serveur Apache et utilisera un système de gestion de base de données de type SQL

# Charte Graphique

Les visuels définis ci-dessous sont disponibles dans le google drive consacré au [projet Quizzy](#)

## I. Logo



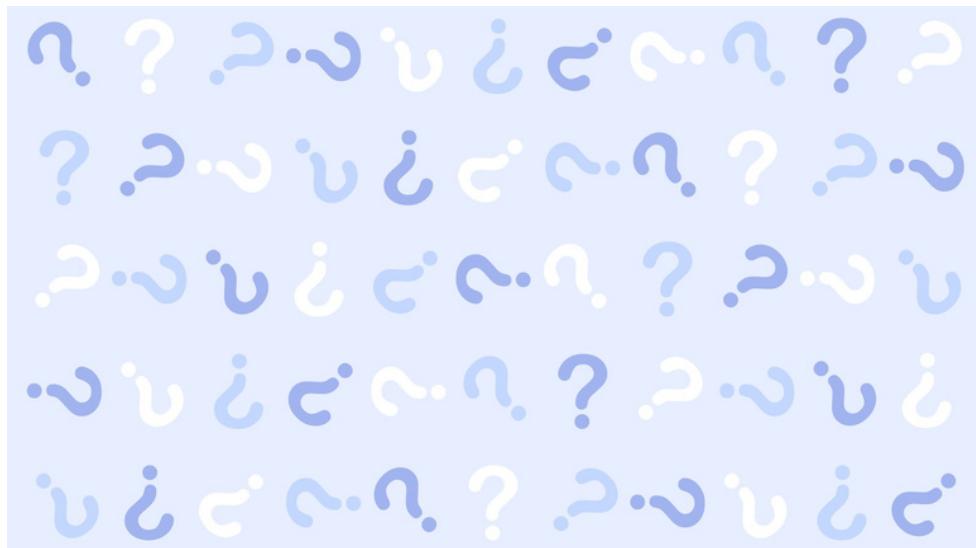
## II. Couleursetfonts

Les couleurs et fonts à utiliser seront les suivantes :

- **Couleur Principale :** #4768d5
- **Couleur Secondaire :** #cfddff
- **Font du "?" servant au "Q" de Quizzy :** Rockwell
- **Font du reste du logo :** Harlow Solid Italic

## III. Background

À l'heure actuelle, seul un background pour device type Desktop - Laptop a été réalisé.



## Minimum Livrable

Afin de commencer à toucher au plus tôt notre cible, il sera possible de mettre en ligne une première version “incomplète” de l’application.

Cette version devra néanmoins permettre les actions suivantes :

- la création de compte
- la participation aux quiz en solo
  - la création de quiz avec des questions de type vrai/faux ou QCM avec des propositions de type texte
- la modification du profil et du mot de passe
- une dizaine de quiz devront être disponibles ainsi qu’une dizaine de catégories

## Délai d'exécution

Afin de pouvoir profiter de la période où nos utilisateurs cibles auront la possibilité de profiter au mieux de leur temps de loisir, l’application devra pouvoir être mise en ligne au travers de son minimum livrable dans un premier temps durant l’été, avant le mois d'Août.

## Environnement de développement

- Gestion de projet :
  - versioning : Github  

  - Tâche (Kanban) : Miro  

- Framework Back End : **Symfony**  

- Framework Front End : **Angular**  

- Base de données : **MySQL**  

- Languages : **PHP, HTML, CSS (Bootstrap & Sass), TypeScript**  

- IDE : **IntelliJ**  

- Hébergement du site : **OVH cloud** (à confirmer)  
